MASTER EN SCIENCES MATHÉMATIQUES

Sur la détermination d'un point admissible en programmation mathématique

FISETTE, Denis

Award date:
1983

Awarding institution:
Universite de Namur

SUR LA DÉTERMINATION D'UN POINT ADMISSIBLE EN PROGRAMMATION MATHÉMATIQUE

présenté par

Denis FISETTE

Promoteur :
Monsieur NGUYEN VAN HIEN

Année 82/83

Que toutes les personnes qui ont contribué à l'élaboration de ce mémoire trouvent ici l'expression de mes remerciements sincères. Je pense entre autres à Messieurs V.H. NGUYEN, Cl. FLEURY, Ch. MERCKX, A. BIHAIN, et tout spécialement à J.J. STRODIOT.

TABLE DES MATIERES

Chapitre V - RESULTATS NUMERIQUES

BIBLIOGRAPHIE

d'un nouveau système d'inéquations, qui diffère aussi légèrement que possible du système initial. Il est donc intéressant de "relaxer les contraintes" en les modifiant le moins possible. Ce sera notre critère d'optimalité pour une "relaxation". Le chapitre IV introduit ce concept de relaxation et propose deux algorithmes. Le premier, basé sur l'intuition seule, servira de référence pour le second qui s'appuie sur une démontration de convergence. Ce second algorithme consiste à appliquer itérativemernt l'algorithme de Schnabel à un système d'inéquations modifié légèrement à chaque itération. Quelques résultats théoriques y sont présentés.

Le chapitre V expose quelques résultats numériques significatifs de l'algorithme de Schnabel et des deux algorithmes proposés au chapitre précédent. Quelques commentaires à leur propos seront exposés en guise de conclusion.

Chapitre I

PRÉSENTATION DE LA MÉTHODE

DE ROBERT B. SCHNABEL

§ 1. Notations et présentation du problème

Soient D un domaine de R^n et c_i, $1 \leqslant i \leqslant m$, m fonctions continues à valeurs réelles définies sur D. Le problème est de déterminer si le système d'inéquations

$$(P) \qquad c_i(x) \leqslant 0 \qquad\qquad i = 1, \ldots, m$$

est compatible, c'est-à-dire si il existe un point x de D satisfaisant chacune des inéquations. Dans le cas contraire, on dira que (P) est incompatible.

Avant de décrire la méthode, il est intéressant de préciser ce concept de compatibilité et de l'illustrer par quelques exemples.

§ 2. Définitions

Définition 1: Soit $\gamma > 0$. Le système d'inéquations (P) est dit strictement γ-compatible si il existe un point $x \in D$ tel que

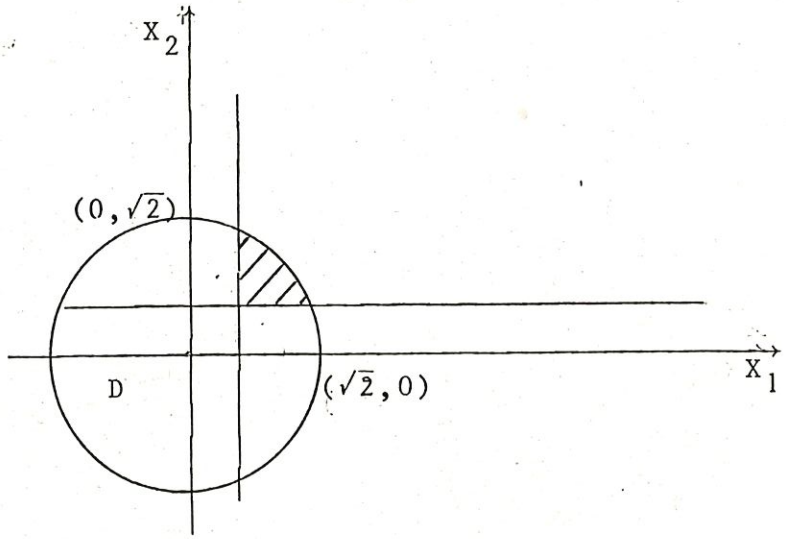
$$c_i(x) \leqslant -\gamma \qquad\qquad i = 1, \ldots m$$

Définition 2: Le système d'inéquations (P) est dit strictement compatible si il existe un nombre $\gamma > 0$ pour lequel il est strictement γ-compatible.

Illustrons ces deux définitions par un exemple simple (contraintes linéaires).

Exemple: Soit D la boule fermée de R^2 centrée en 0 et de rayon $\sqrt{2}$ et considérons le système

$$(P^1) \begin{cases} -x_1 + \frac{1}{2} \leqslant 0 \\ -x_2 = \frac{1}{2} \leqslant 0 \end{cases}$$



(P^1) est strictement compatible car il est par exemple strictement $\frac{1}{2}$ - compatible. Il suffit de prendre $x_1 = 1$ et $x_2 = 1$. Par contre, (P^1) n'est pas strictement 3/2-compatible.

Définition 3: Le système d'inéquations (P) est dit exactement compatible si il est compatible mais pas strictement.

Exemples: 1) Soit $D = \bar{B}(0, \sqrt{2})$ et considérons

$$(P^2) \begin{cases} -x_1 + 1 \leqslant 0 \\ -x_2 + 1 \leqslant 0 \end{cases}$$

(P^2) est exactement compatible. En effet, le point $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ est compatible, cependant, $\forall\ \gamma > 0$, le système

$$\begin{cases} -x_1 + 1 \leqslant -\gamma \\ -x_2 + 1 \leqslant -\gamma \end{cases}$$

est incompatible.

2) Soient D = R et le système d'inéquations

$$(P^3) \quad \begin{cases} c_1(x) = 1 - x^3 \leqslant 0 \\ c_2(x) = x^2 - 1 \leqslant 0 \end{cases}$$

(P^3) est compatible en 1 mais, $\forall\ \gamma > 0$

$$\begin{cases} 1 - x^3 \leqslant -\gamma \\ x^2 - 1 \leqslant -\gamma \end{cases}$$

est incompatible.

Définition 4: Soit $\gamma > 0$. Le système d'inéquations (P) est dit compatible à γ-près si (P) est incompatible et si le système

$$c_i(x) - \gamma \leqslant 0 \qquad i = 1, \ldots, m$$

est compatible.

Définition 5: Le système d'inéquations (P) est dit strictement incompatible si il existe un nombre $\gamma > 0$ pour lequel (P) n'est pas compatible à γ-près.

Définition 6: Le système d'inéquations (P) est dit exactement incompatible si il est incompatible mais pas strictement ou si (P) est incompatible mais pout tout $\gamma > 0$, le système

$$c_i(x) - \gamma \leqslant 0 \qquad i = 1, \ldots, m$$

est compatible.

Exemple: Soit $D = \bar{B}(0, \sqrt{2})$. Le système

$$(P^4) \quad \begin{cases} -x_1 + 5/4 \leqslant 0 \\ -x_2 + 5/4 \leqslant 0 \end{cases}$$

est compatible à $\frac{1}{4}$ - près mais strictement incompatible car (P^4) n'est pas compatible à $\frac{1}{20}$ - près.

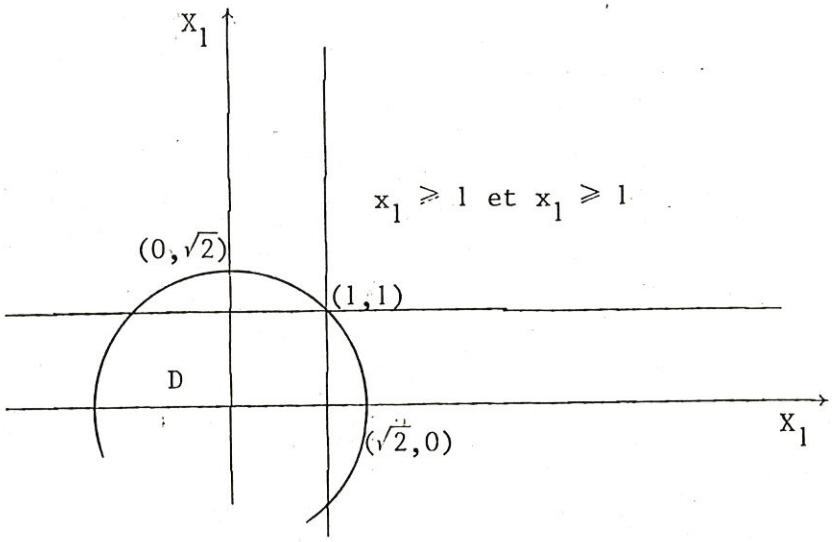
Proposition 1: Si D est fermé et borné, le système d'inéquations (P) ne peut pas être exactement incompatible.

Exemples: a) (utilité du caractère fermé)

Soit D, la boule ouverte de R^2 centrée en 0 et de rayon $\sqrt{2}$.

Le système

$$(P^6)\begin{cases} x_1 \geqslant 1 \\ x_2 \geqslant 1 \end{cases}$$

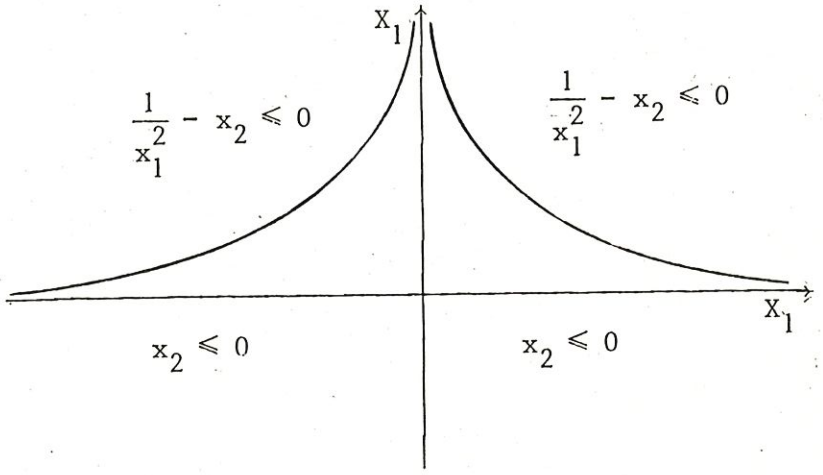


est exactement incompatible.

b) (utilité du caractère borné)

Soit D = R^2 et considérons le système

$$(P^7)\begin{cases} \frac{1}{x_1^2} - x_2 \leqslant 0 \\ x_2 \leqslant 0 \end{cases}$$



qui est également exactement incompatible.

Démonstration de la proposition 1

Supposons par l'absurde que le système (P) est exactement incompatible. Par définition, pour tout entier j, le système

$$c_i(x) - \frac{1}{j} \leqslant 0 \qquad i = 1, \ldots, m$$

est compatible. Par conséquent, pour chaque j, il existe $x_j \in D$ satisfaisant chacune des inégalités précédentes. Comme D est fermé et borné, il existe une sous-suite de $(x_j)_j$ convergeant vers un point $x^{\star} \in D$. D'autre part, c_i étant continu, on obtient par passage à la limite sur j que

$$c_i(x^{\star}) \leqslant 0 \qquad i = 1, \ldots, m$$

c'est-à-dire le système (P) est compatible. On obtient ainsi une contradiction et la propriété est démontrée.

□

§ 3. Une méthode de pénalisation

3.1. Notion fondamentale

La méthode de pénalisation de Schnabel, comme son nom l'indique, est basée sur l'utilisation d'une fonction pénalisante.

Définition: Une fonction pénalisante est une fonction $\omega \in C^2$

$$\omega : R \longrightarrow R$$

qui vérifie (i)$\longrightarrow$(iv)

(i) $\omega(0) = 0$

(ii) $\omega'(y) > 0 \qquad \forall\, y \in R$

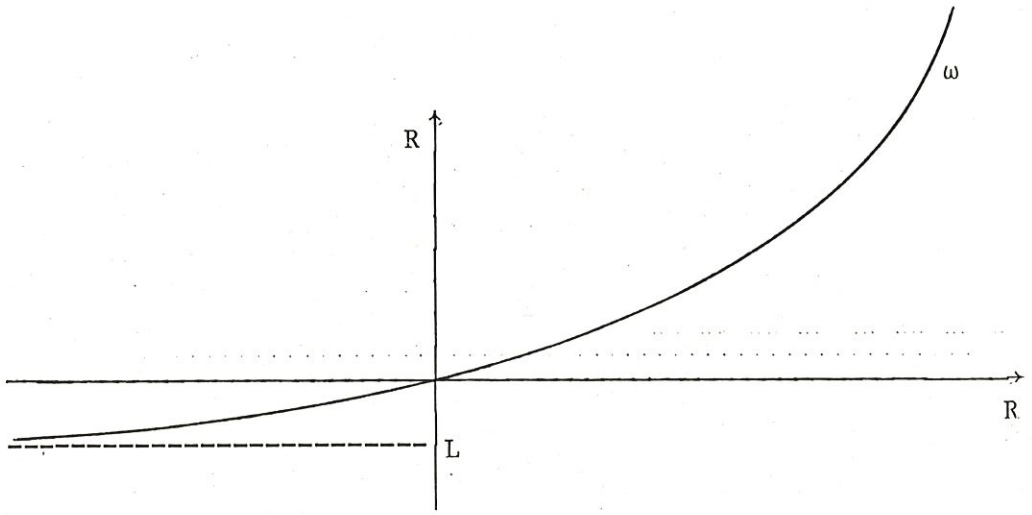
(iii) $\omega''(y) > 0 \qquad \forall\, y \in R$

(iv) $\lim_{y \to -\infty} \omega(y) = L \qquad$ avec $L > -\infty$

De plus, (i)$\longrightarrow$ (iv) nous donne

$$(v) \quad \lim_{y \to +\infty} \omega(y) = +\infty$$

Voici le graphe type d'une fonction pénalisante



3.2. Exemples de fonctions pénalisantes

1) $e^y - 1 \qquad \forall\, y \in R$

2) $$\begin{cases} e^y - 1 & y \leqslant \alpha \\ \frac{1}{2}(y-\alpha)^2 e^\alpha + (y-\alpha)e^\alpha + (e^\alpha - 1) & y > \alpha \end{cases}$$

avec α paramètre strictement positif

3) $$\begin{cases} \frac{y}{1-y} & y \leqslant \beta \\ \frac{(y-\beta)^2}{(1-\beta)^3} + \frac{(y-\beta)}{(1-\beta)^2} + \frac{\beta}{(1-\beta)} & y > \beta \end{cases}$$

avec $\beta \in]0,1[$

Toutes les notions utiles dans l'algorithme ont été définies. Voyons dès lors la méthode proprement dite.

3.3. Idée de base

Formulons de façon générale le problème que l'on se propose de résoudre.

Soit $D \subset R^n$ et soient $c_i : D \longrightarrow R \qquad i = 1, \ldots, m$ des fonctions que l'on suppose de classe C^2. Considérons le système d'inéquations suivant:

$$(P) \qquad c_i(x) \leqslant 0 \qquad\qquad i = 1, \ldots, m \qquad\qquad (1.0)$$

et définissons pour chaque valeur de $p \geqslant 0$ la fonction

$$\phi(x,p) = \frac{1}{p} \sum_{i=1}^{m} \omega(p.c_i(x))$$

où ω est une fonction pénalisante.

La méthode consistera à calculer pour un certain nombre de valeurs du paramètre p, la solution du problème

$$\min_{x \in D} \ \phi(x,p) \qquad\qquad (1.1)$$

Les valeurs de p seront choisies de telle sorte que l'on trouve un point compatible ou que l'on puisse affirmer que le système est incompatible.

En augmentant la valeur de p, les contraintes violées vont être de plus en plus pénalisées dans la minimisation de $\phi(x,p)$. Pour cela, il suffit de considérer les propriétés (i) à (iii) de la définition de ω. Par contre, les contraintes satisfaites vont être "récompensées" mais nettement moins car, de toute façon, ω est bornée inférieurement par une constante finie L en vertu de (iv).

Nous sommes maintenant en mesure de décrire l'algorithme.

3.4. Algorithme I

INITIALISATION. Déterminer $p_o > 0$, $p^\star > 0$ ainsi que x_o point de départ.
Poser $p = p_o$, $x = x_o$ et aller à l'étape principale.

ETAPE PRINCIPALE

1 Rechercher le minimum de $\phi(x,p)$ sur $D \subset R^n$. Noter $\phi^\star(p)$ la valeur optimale et $x^\star(p)$ la solution optimale obtenue.
Si $c_i(x^\star(p)) \leqslant 0$ $\quad i = 1, \ldots, m,$ aller en # 2
sinon aller en # 3.

2 Le système d'inéquations (P) est compatible: $x^\star(p)$ est un point compatible.
STOP

3 Si $\phi^\star(p) > 0$ aller en # 4
sinon poser $p = p^\star + p$ et aller en # 1

4 Le système d'inéquations (P) est incompatible.
STOP

Remarque 1: Dans la suite de ce chapitre, $x^\star(p)$ est considéré comme étant le minimum théorique (i.e. global) de la phase de minimisation en # 1. L'obtention d'un minimum local peut provoquer certains problèmes: ce sera entre autres l'objet du chapitre III.

Remarque 2: Si $\phi^\star(p)$ est strictement positif, on détecte l'incompatibilité du système (P). En effet, supposons le système (P) compatible. Il existe donc $x_o \in D$ tel que

$$c_i(x_o) \leqslant 0 \qquad i = 1, \ldots, m.$$

et dès lors, la fonction $\phi(x,p)$ évaluée au point x_o sera négative

$$\phi(x_o, p) \leqslant 0 < \phi^\star(p)$$

ce qui contredit le caractère optimal de $\phi^\star(p)$.

□

Remarque 3: La méthode consiste donc à incrémenter le paramètre de pénalisation p jusqu'à ce que l'on détecte soit la compatibilité du système (P) ainsi qu'un point compatible, soit son incompatibilité. Intuitivement, nous voyons que si le système (P) est strictement compatible, alors "$x^{\star}(p)$" va avoir tendance à vérifier toutes les contraintes car celles qui resteront positives vont être pénalisées de plus en plus en incrémentant le paramètre "p". Si, par contre, le système (P) est strictement incompatible, à chaque minimisation, au moins une contrainte restera strictement positive et ce terme, en augmentant le paramètre de pénalisation, prévaudra face aux autres et dès lors, $\phi^{\star}(p)$ sera strictement positif.

Voici quelques résultats théoriques correspondant aux remarques précédentes :

a) $\phi^{\star}(p)$ est une suite croissante pour p croissant;

b) si le système d'inéquations (P) est strictement compatible, alors, lorsque p atteint ou dépasse une valeur "critique" finie, tout $x^{\star}(p)$ fourni par l'algorithme est compatible;

c) si, par contre, le système (P) est strictement incompatible, on montre que lorsque p atteint ou dépasse une valeur finie, la valeur $\phi^{\star}(p)$ devient positive et le reste en vertu du premier résultat.

Restent cependant deux cas possibles, l'exacte compatibilité et l'exacte incompatibilité, qui seront traités plus tard.

§ 4. Résultats théoriques

Pour démontrer ces résultats, voyons d'abord un lemme de la théorie des fonctions convexes.

Lemme 1: Soit $\omega : R \longrightarrow R$, une fonction de classe C^1 strictement convexe telle que $\omega(0) = 0$.

Soit, de plus, $p^+ > p > 0$. Dès lors, pour tout réel y, on a

$$\frac{\omega(p^+y)}{p^+} \geqslant \frac{\omega(py)}{p}$$

avec l'égalité ssi $y = 0$.

Démonstration: De deux choses l'une :

$y = 0$ et l'on a trivialement la thèse

ou $y \neq 0$.

Dans ce cas, prenons $t = p/p^+$ et donc t appartient à l'intervalle $]0,1[$. La définition de la convexité stricte nous donne :

$$\omega(p.y) = \omega(tp^+y + (1-t).0) < t\omega(p^+y) + (1-t)\,\omega(0)$$

$$= p/p^+ \;\; \omega(p^+y)$$

□

Etablissons maintenant la croissance des valeurs $\phi^{\star}(p)$.

Théorème 1: Soit $\omega : R \longrightarrow R$, satisfaisant les hypothèses du lemme 1. Soient de plus, p et p^+ tels que $p^+ > p > 0$. Considérons le problème (1.0) et gardons les notations habituelles de l'algorithme I. Alors

$$\phi^{\star}(p^+) \geqslant \phi^{\star}(p)$$

avec l'égalité seulement si $c_i[x^{\star}(p^+)] = 0 \qquad i = 1, \ldots, m.$

Démonstration: En vertu du lemme 1,

$$\frac{\omega(p^+ . c_i(x^*(p^+)))}{p^+} \geqslant \frac{\omega(p . c_i(x^*(p^+)))}{p}$$

inégalité valable pour $i = 1, \ldots, m$ et avec l'égalité ssi $c_i(x^*(p^+)) = 0 \quad i = 1, \ldots, m$. Sommons sur toutes les contraintes pour obtenir

$$\phi^*(p^+) \geqslant \phi(x^*(p^+), p)$$

avec égalité ssi $c_i(x^*(p^+)) = 0 \quad i = 1, \ldots, m.$

Mais par définition de $\phi^*(p)$, nous obtenons la thèse puisque

$$\phi^*(p^+) \geqslant \phi(x^*(p^+), p) \geqslant \phi^*(p)$$

□

Venons-en maintenant aux résultats concernant la convergence de l'algorithme face à des systèmes d'inéquations strictement compatibles ou strictement incompatibles.

Théorème 2: Soit ω vérifiant les conditions (i), (ii) et (iv) d'une fonction pénalisante et considérons le problème (1.0). Si le système d'inéquations (P) est strictement compatible, alors il existe $p_1 \geqslant 0$ tel que, pour tout $p \geqslant p_1$, le point $x^*(p)$ fourni par l'algorithme est compatible.

Démonstration: (P) est strictement compatible, ce qui, par définition, équivaut à dire qu'il existe $\gamma > 0$ et $x_1 \in D$ tels que $c_i(x) \leqslant -\gamma \quad i = 1, \ldots, m.$

Choisissons $y_1 < 0$ tel que

$$\omega(y_1) \leqslant \frac{(m-1)L}{m}$$

et posons $p_1 = -y_1/\gamma$

Supposons par l'absurde que pour $p \geqslant p_1$, $x^{\star}(p)$ ne soit pas compatible. Il existe alors au moins une contrainte violée i.e.

$$\exists\, i_o \text{ tel que } c_{i_o}(x^{\star}(p)) > 0$$

et donc, en vertu des propriétés (i) et (iv) des fonctions pénalisantes, on a les inégalités suivantes:

$$\omega(p.c_{i_o}(x^{\star}(p))) > 0$$

$$\phi_{\star}(p) > \frac{(m-1)L}{p} \qquad (1.2)$$

D'autre part,

$$\phi(x_1, p) \leqslant m\,\frac{\omega(-p\gamma)}{p} \qquad (1.3)$$

et par la monotonie de ω

$$\omega(-p\gamma) \leqslant \omega(-p_1\gamma) = \omega(y_1) \leqslant \frac{(m-1)L}{m}$$

majorant $\omega(-p\gamma)$ par $\frac{(m-1)L}{m}$ dans (1.3), nous avons

$$\phi^{\star}(p) \leqslant \phi(x_1, p) \leqslant \frac{(m-1)L}{p}$$

en contradiction avec (1.2).

□

Le théorème 1 nous montre donc que, face à un système (P) strictement compatible, l'algorithme détecte un point compatible. Etablissons maintenant un résultat concernant l'incompatibilité stricte du système (P).

Théorème 3: Soit $\omega : R \longrightarrow R$ une fonction pénalisante et considérons le problème (1.0). Supposons le système d'inéquations (P) strictement incompatible. Alors il existe $p_2 \geqslant 0$ tel que, pour tout $p \geqslant p_2$, $\phi^{\star}(p)$ est strictement positif.

Démonstration: Par définition de l'incompatibilité stricte du système (P), il existe $\delta > 0$ tel que (P) ne soit pas compatible à ce δ-près.

(1.4)

Soit $y_2 > 0$, choisi tel que

$$\omega(y_2) \geq -(m-1)L$$

et posons $p_2 = y_2/\delta$. La thèse revient donc à montrer que pour tout $p \geq p_2$, $\phi^{\star}(p)$ est positif.

Or, (1.4) nous donne:

$$\forall\, x \in D \;\exists\, i \text{ tel que } c_i(x) \geq \delta$$

d'où, par (iv) et (ii) d'une fonction pénalisante

$$\phi^{\star}(p) > \frac{[(m-1)L + \omega(p.\delta)]}{p}$$

De plus, par définition de p_2 et la monotonie de ω,

$$\omega(p\delta) \geq \omega(p_2\delta) = \omega(y_2) \geq -(m-1)L \qquad (1.6)$$

Combinant (1.5) et (1.6), nous avons la thèse:

$$\phi^{\star}(p) > 0$$

□

Nous venons donc de voir que dans le cas de la compatibilité stricte ou de l'incompatibilité stricte, l'algorithme est fonctionnel puisqu'il détecte et peut affirmer la stricte compatibilité ou incompatibilité dès l'obtention d'un paramètre p plus grand que le "p critique" qui est fini en vertu des théorèmes 2 et 3.

De plus, on a vu précédemment que l'on pouvait éviter l'exacte incompatibilité avec des conditions sur D. Reste donc à envisager la compatibilité exacte. Or dans ce dernier cas, on peut montrer que, pour tout $\delta > 0$, lorsque p atteint ou dépasse une valeur p_δ finie, chaque $x^{\star}(p)$ est compatible à ce δ-près. Un critère d'arrêt est donc nécessaire mais démontrons d'abord cette propriété.

Proposition 2: Soit ω une fonction pénalisante et considérons le problème (1.0) où le système (P) est supposé exactement compatible. Alors, pour tout $\delta > 0$, il existe une valeur finie p_δ telle que, pour tout $p \geqslant p_\delta$, $x^\star(p)$ est compatible à ce δ-près.

Démonstration: Soit δ fixé, $\delta > 0$ et considérons

$$y_\delta \text{ tel que } \omega(y_\delta) > -(m-1)L$$

posons $p_\delta = y_\delta / \delta$ et, par l'absurde, supposons que pour $p \geqslant p_\delta$, $x^\star(p)$ n'est pas compatible à ce δ près. Il existe donc une contrainte violée de plus de δ, i.e.

$$\exists\ i \text{ tel que } c_i(x^\star(p)) > \delta$$

et donc

$$\phi^\star(p) > \frac{(m-1)L}{p} + \frac{\omega(\delta p)}{p} \qquad (1.7)$$

Or, par la monotonie de ω,

$$\omega(p\delta) \geqslant \omega(p_\delta\ \delta) = \omega(y_\delta) \geqslant -(m-1)L$$

Minorant $\omega(p_\delta)$ par $-(m-1)L$ dans (1.7), nous avons

$$\phi^\star(p) > 0 \qquad (1.8)$$

D'autre part, (P) étant exactement compatible, il existe x_o tel que $c_i(x_o) \leqslant 0 \qquad i = 1, \ldots, m$ et donc

$$\phi^\star(p) \leqslant \phi(x_o, p) \leqslant 0$$

en contradiction avec (1.8). □

Un critère d'arrêt est cependant nécessaire pour l'utilisation sur ordinateur de cet algorithme.

§ 5. Algorithme général

5.1. Algorithme II

INITIALISATION Déterminer $p_o \geqslant 0$ et x_o point de départ.
Choisir un $\varepsilon > 0$.
Poser $x = x_o$, $p = p_o$ et aller à l'étape principale.

ETAPE PRINCIPALE

1 Rechercher $\phi^{\star}(p)$ et $x^{\star}(p)$.
Si $\phi^{\star}(p) > 0$, aller en # 2,
sinon si $\max_{1 \leqslant i \leqslant m} c_i(x^{\star}(p)) < \varepsilon$, aller en # 3
sinon aller en # 4.

2 Les contraintes sont incompatibles.
STOP

3 Les contraintes sont compatibles (☆)
STOP

4 Incrémenter le paramètre de pénalisation p.
Aller en # 1.

Voyons que le critère d'arrêt choisi fonctionne bien puisque, sans aucune condition sur la compatibilité du système d'inéquations(P), on est assuré de l'arrêt de l'algorithme par le théorème suivant:

(☆) Remarquons que l'algorithme peut sortir le message de l'étape # 3 pour des contraintes compatibles à un δ-près, $\delta < \varepsilon$.

Théorème 4: Soit ω: R$\longrightarrow$R une fonction pénalisante et considérons le problème (1.0). Alors, étant donné un $\varepsilon > 0$, il existe $p_3 \geqslant 0$ tel que l'on ait soit $\phi^{\star}(p) > 0$, soit un point $x^{\star}(p)$ compatible à un ε-près pour tout $p \geqslant p_3$.

Démonstration: Soit $y_2 > 0$ tel que $\omega(y_2) \geqslant -(m-1)L$.

Posons

$$\delta = \frac{\varepsilon}{m+1} \qquad \text{et} \qquad p_3 = \frac{y_2}{\delta} .$$

Si le système des inéquations (P) n'est pas compatible à un δ-près, la démonstration du théorème 3 nous assure l'inégalité suivante:

$$\phi^{\star}(p) > 0 \qquad \text{pour } p \geqslant p_3.$$

Considérons maintenant l'autre possibilité: (P) est compatible ou compatible à un δ-près. Il existe donc un point $x_3 \in D$ tel que

$$c_i(x_3) \leqslant \delta \qquad i = 1, \ldots, m$$

et dès lors, pour tout $p \geqslant 0$

$$\phi(x_3,p) \leqslant m\ \omega(p.\delta)\ /p \qquad (1.11)$$

Supposons par l'absurde que pour $p \geqslant p_3$, $x^{\star}(p)$ n'est pas compatible à ε-près. Dès lors, par définition des fonctions pénalisantes, (ii) et (iv), nous avons

$$\phi^{\star}(p) > \frac{(m-1)L + \omega(p.\varepsilon)}{p} \qquad (1.9)$$

De plus, par le lemme 1

$$\omega(p.\varepsilon) \geqslant (m+1)\ \omega\left(\frac{p.\varepsilon}{m+1}\right) = (m+1)\ \omega(p.\delta)$$

ce qui, dans l'inégalité (1.9) nous donne

$$\phi^{\star}(p) > \frac{[(m-1)L + \omega(p.\delta) + m\ \omega(p.\delta)]}{p} \qquad (1.10)$$

et par la monotonie de ω

$$\omega(p\delta) \geqslant \omega(p_3\delta) = \omega(y_2) \geqslant -(m-1)L$$

que l'on remplace dans (1.10) et l'on obtient

$$m.\frac{\omega(p\delta)}{p} < \phi^{\star}(p)$$

en contradiction avec (1.11).

□

Nous avons donc démontré que l'algorithme, face à des contraintes quelconques de type inégalité peut décerner la compatibilité ou l'incompatibilité de celles-ci, pour autant que l'on puisse calculer $x^{\star}(p)$ qui doit être un minimum global. Or, on est assuré de ce caractère global de l'optimum seulement si les contraintes sont convexes. De plus, la majorité des routines de minimisation nécessite une fonction objectif de classe C^1 (ou C^2). C'est pourquoi, on imposera des hypothèses supplémentaires sur les contraintes suivant les exigences de la sous-routine de minimisation utilisée.

§ 6. Résultat intéressant

Dans la plupart des problèmes pratiques, il est intéressant de trouver un point vérifiant les contraintes avec "une marge de sécurité aussi grande que possible". Traduisons cette notion mathématiquement. On considère le problème (1.0) que l'on suppose strictement compatible.

$$\text{Notons} \qquad k^{\star} = -\operatorname*{Min}_{x \in D}\left[\operatorname*{Max}_{1 \leqslant i \leqslant m} c_i(x)\right]$$

c'est-à-dire $k^{\star}$ est la plus grande valeur de "k" pour laquelle le système (P) est strictement "k"-compatible.

Un résultat fort intéressant consiste à montrer que, pour certaines valeurs de k, $k < k^{\star}$, l'algorithme peut trouver un point $x \in D$ tel que

$$c_i(x) \leqslant -k \qquad i = 1, \ldots, m.$$

A ce propos, on verra l'importance du choix de la fonction pénalisante.

Théorème 5: Soit ω: $R \longrightarrow R$, une fonction pénalisante et définissons $\hat{\omega}(y) = \omega(y) - L$. Considérons le problème (1.0) où le système (P) est supposé strictement $k^{\star}$-compatible. Soit, de plus, k tel que $0 < k < k^{\star}$.

Faisons l'hypothèse supplémentaire qu'il existe $p_4 \geqslant 0$ tel que

$$\forall\ p \geqslant p_4, \quad m\hat{\omega}(-pk^{\star}) \leqslant \hat{\omega}(-pk) \qquad (1.12)$$

Alors, pour tout $p \geqslant p_4$, chaque $x^{\star}(p)$ est strictement k-compatible.

Démonstration: Par l'absurde, supposons que, pour $p \geqslant p_4$, $x^{\star}(p)$ n'est pas strictement k-compatible. Dès lors, une des contraintes au moins a une valeur supérieure à (-k). On peut donc écrire :

$$\phi^{\star}(p) > \frac{(m-1)L + \omega(-pk)}{p} = \frac{m.L + \hat{\omega}(-pk)}{p} \qquad (1.13)$$

Parallèlement, puisque le système des inéquations est strictement $k^{\star}$-compatible, il existe $x_4 \in D$ tel que $c_i(x_4) \leqslant -k^{\star}$, $i = 1, \ldots, m$. C'est pourquoi, en vertu de la monotonie de ω, nous avons

$$\phi^{\star}(p) \leqslant \phi(x_4, p) \leqslant \frac{m.\omega(-pk^{\star})}{p} = \frac{m.L + m\,\hat{\omega}(-pk^{\star})}{p} \qquad (1.14)$$

et l'on termine la démonstration en remarquant que (1.13) et (1.14) sont en contradiction avec l'hypothèse (1.12).

□

Corollaire 1: Soient $\omega(y) = e^y - 1$ pour $y < 0$ et k tel que $0 < k < k^{\star}$. Supposons que le système (P) du problème (1.0) est strictement $k^{\star}$-compatible. Dès lors, il existe p_5 tel que, $\forall\ p \geqslant p_5$, $x^{\star}(p)$ est strictement k-compatible.

Démonstration: Regardons pour quelles valeurs de "k" l'hypothèse (1.12) du théorème précédent est vérifiée. Or,

$$\hat{\omega}(y) = e^y$$

et donc l'hypothèse (1.12) se récrit :

$\exists\ p_5$ tel que, pour $p \geqslant p_5$, $m\, e^{-pk^{\star}} \leqslant e^{-pk}$

et nous voyons que, quel que soit k, $k < k^{\star}$, en choisissant p_5 comme suit,

$$p_5 = \frac{\ln m}{(k^{\star} - k)}$$

on a l'inégalité (1.12), ce qui termine la preuve en vertu du théorème précédent.

□

Remarque: Si, par contre, $\omega(y) = \frac{y}{1-y}$ pour $y < 0$

alors

$$\hat{\omega}(y) = \frac{1}{1-y}$$

et donc, l'hypothèse (1.12) nécessite

$$\frac{m}{(1+pk^{\star})} \leqslant \frac{1}{(1+pk)}$$

ou encore

$$(m-1) \leqslant p.(k^{\star} - mk)$$

qui est possible seulement si $k < \frac{k^{\star}}{m}$. Dès lors, avec cette fonction

de pénalisation et face à un système d'inéquations strictement $k^{\star}$-compatible, le théorème précédent garantit seulement que, pour p suffisamment grand, les points fournis par l'algorithme sont strictement $(k^{\star}/m - \varepsilon)$-compatibles.

§ 7. Et sur ordinateur...

Schnabel présente quelques résultats obtenus sur ordinateur. Il a implémenté cet algorithme avec la fonction de pénalisation $e^{y} - 1$, en traitant séparément les contraintes linéaires $(l_q(x))$ et non linéaires $(c_i(x))$. La séquence des sous-problèmes de minimisation devient

$$\begin{cases} \underset{x \in D}{\text{Min}} \quad \phi(x,p) = \dfrac{1}{p} \displaystyle\sum_{i=1}^{m} \left(e^{p \cdot c_i(x)} - 1\right) \\ \\ \text{sous les contraintes} \\ \\ l_q(x) \leqslant 0 \qquad\qquad q = 1, \ldots, t \end{cases}$$

La sous-routine d'optimisation utilisée est une variante de l'algorithme de Fletcher et Goldfarb qui contient une stratégie de "restart" pour "augmenter", dit-il, "les chances de trouver un minimum global".

Il est en effet très important de trouver le vrai minimum pour le bon fonctionnement de cet algorithme.

Restent cependant deux problèmes (plus ou moins arbitraires) :

1) le choix du paramètre de pénalisation de départ;
2) son incrémentation.

Schnabel propose d'initialiser p à 0 car on montre facilement par l'Hospital que

$$\phi(x,0) \triangleq \lim_{p \to o} \phi(x,p) = \sum_{i=1}^{m} c_i(x)$$

et il incrémente le paramètre p suivant une méthode non explicitée dans cet article. Il précise cependant que la nouvelle valeur du paramètre appartient à l'intervalle [2p,10p] sauf si p = 0 bien sûr.

Il présente les résultats de son programme face à des contraintes relativement simples. On peut trouver ceux-ci pages 145 et 146 de la revue

§ 8. Modifications possibles

Une amélioration de cet algorithme a été proposée: elle consiste à changer le sous-problème de minimisation en

$$\min_{x \in D} \frac{1}{p} \sum_{i=1}^{m} \lambda_i(p)\ \omega(p.c_i(x))$$

modifiant l'estimation du multiplicateur de Lagrange $\lambda(p)$ par la formule

$$\lambda_i(p^+) = \lambda_i(p).\ \omega'[p.c_i(x^*(p))]$$

pour l'itération suivante. (Réf. [4]).

Cette tentative d'amélioration provient très vraisemblablement des méthodes du Lagrangien augmenté. Mais, l'utilisation des $\lambda_i(p)$", estimations des multiplicateurs, est moins justifiée vu l'absence de fonction objectif. De plus, l'utilité première des méthodes de pénalisation est de ne plus devoir faire tendre le paramètre p vers l'infini; propriété que l'on a justement pu démontrer dans le cadre de notre problème. Schnabel ajoute qu'il n'est pas évident que les propriétés démontrées dans cet article resteront valides pour ce changement.

Chapitre II

PRÉSENTATION D'UNE CLASSE DE PROBLÈMES PRATIQUES

Un type de problèmes de minimisation fréquemment rencontrés par le laboratoire d'aéronautique de l'université de Liège peut se formuler de la façon suivante :

$$\begin{cases} \underset{x \in R^n}{\text{Min}} \quad \sum_{i=1}^{n} \alpha_i x_i \\ \text{s.c.} \quad b_j \leqslant \sum_{i=1}^{n} \frac{a_{ij}}{x_i} \leqslant c_j \qquad j = 1, \ldots, m \end{cases}$$

Ces problèmes sont des approximations successives d'un problème de minimisation très coûteux à résoudre directement. En effet, une évaluation de la fonction objectif nécessite une recherche par éléments finis. (Pour plus d'informations, consulter [6] et [7]).

La résolution de cette classe de problèmes est assez simple. En effet, après avoir effectué le changement de variables $t_i = \frac{1}{x_i}$, $i = 1, \ldots, n$, ces problèmes reviennent à la minimisation d'une fonction linéaire en $(\frac{1}{t_i})$ sous contraintes linéaires et, dès lors, la détermination d'un point admissibme de départ est aisée (Simplex). Mais l'addition de contraintes linéaires a rendu plus difficile la résolution de ces problèmes. En effet, ceux-ci sont devenus :

$$\underset{x \in R^n}{\text{Min}} \begin{cases} \sum_{i=1}^{n} \alpha_i x_i \\ \text{s.c.} \quad b_j \leqslant \sum_{i=1}^{n} \frac{a_{ij}}{x_i} \leqslant c_j \\ d_k \leqslant \sum_{i=1}^{n} l_{ik} x_i \leqslant f_k \end{cases}$$

Remarquons que le changement de variables utilisé auparavant n'est plus efficace. Il est dès lors intéressant de vérifier si ces contraintes sont compatibles. De plus, si c'est compatible, l'algorithme de minimisation utilisé par après nécessite un point de départ admissible. Dans le cas contraire, une relaxation des contraintes est nécessaire.

C'est à ce type de problèmes que je ferai référence dans les chapitres suivants. La résolution numérique de certains d'entre eux est présentée au chapitre V.

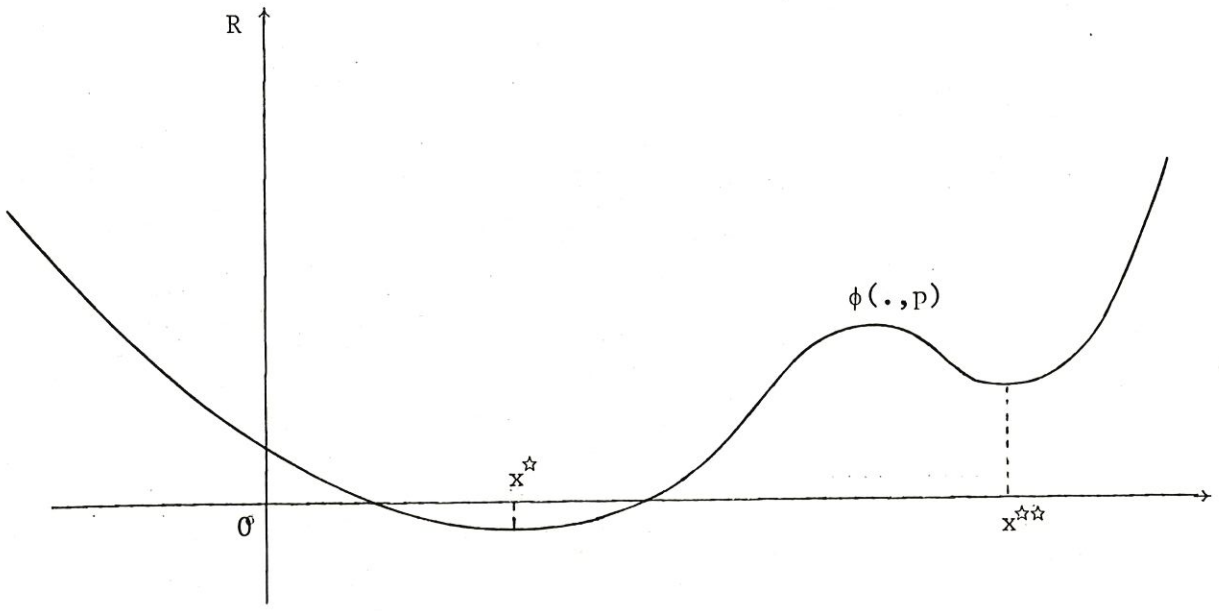
Chapitre III

COMMENTAIRES ET CRITIQUES

§ 1..Quant à la non-convexité...

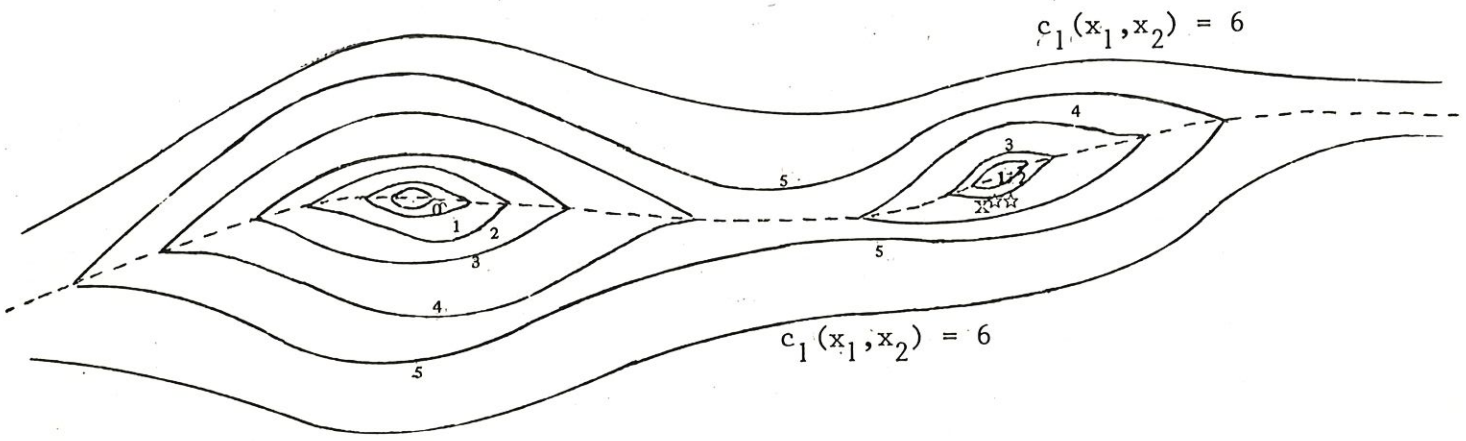
Tout l'article de Schnabel est basé sur l'obtention d'un minimum global $x^{\star}(p)$ de la fonction objective $\phi(x,p)$. Malheureusement, si l'on doit déterminer le caractère compatible ou non d'un ensemble de contraintes non convexes, on n'est plus assuré du caractère global de l'optimum et cela, quelle que soit la sous-routine de minimisation employée. A ce propos, Schnabel dit qu'il utilise une sous-routine avec procédure de "restart", pour "augmenter les chances" de trouver un minimum global.

On comprend facilement l'importance de trouver le vrai minimum. En effet, la sortie "Incompatibilité" de l'algorithme est tout à fait faussée si le minimum détecté $x^{\star\star}$ est local puisque $\phi(x^{\star\star},p)$ peut être positif alors qu'il existe des points compatibles, par exemple $x^{\star}$ sur ce graphe :



Ce problème semble insoluble mais remarquons que celui-ci n'est pas propre à notre algorithme puisque, face à un ensemble de contraintes non convexes, les autres méthodes de résolution rencontrent le même type de difficultés. Sur l'exemple suivant, les méthodes dites "Minimax" partant d'un point voisin de $x^{\star\star}$ décerneraient l'incompatibilité (pour ces méthodes, consulter par exemple [5]).

Soient les courbes de niveau de $\underset{i=1,2}{\text{Max}}(c_i(x))$



Heureusement, dans certains problèmes pratiques tel celui exposé au chapitre II, on a dès le départ une idée de la région dans laquelle devraient se trouver des points admissibles, si il y en a, bien évidemment. Dans les autres cas, la seule issue possible me semble être celle préconisée par Schnabel, c'est-à-dire l'emploi d'une sous-routine avec procédure de restart. Malgré cela, la sortie "incompatibilité" reste incertaine si les contraintes ne sont pas convexes. Cependant, considérons le théorème IV du premier chapitre. Dans un but pratique, démontrons-le sans utiliser le caractère global de chaque $x^{\star}(p)$, c'est-à-dire vérifions que le critère d'arrêt reste valable même si les contraintes ne sont pas convexes.

Proposition 3: Sous les hypothèses et notations du théorème IV, le critère d'arrêt proposé reste fonctionnel même si les contraintes ne sont pas convexes.

Rappel : Critère d'arrêt utilisé: $\phi^{\star}(p) > 0$ ou $\underset{1 \leqslant i \leqslant m}{\text{Max}} c_i(x^{\star}(p)) \leqslant \varepsilon$

Démonstration: Considérons y_6 tel que $\omega(y_6) \geqslant -(m-1)L$ et notons $p_6 = y_6/\varepsilon$. Pour $p \geqslant p_6$, voyons que l'algorithme s'arrête. En effet, de deux choses l'une: soit

$$\underset{1 \leqslant i \leqslant m}{\text{Max}} c_i(x^{\star}(p)) \leqslant \varepsilon$$

et l'algorithme arrête en vertu du critère, soit

$$\underset{1 \leqslant i \leqslant m}{\text{Max}} c_i(x^{\star}(p)) > \varepsilon$$

et, dans ce cas, voyons que $\phi^{\star}(p) > 0$. En effet :

$$\phi^{\star}(p) > \frac{(m-1)L}{p} + \frac{1}{p}\,\omega(p.\varepsilon)$$

Or, par la monotonie de ω

$$\omega(p.\varepsilon) \geqslant \omega(p_6.\varepsilon) > -(m-1)L$$

et donc, $\phi^{\star}(p) > 0$. Ceci termine la démonstration puisque l'on est assuré de l'arrêt de l'algorithme dès que p atteint ou dépasse p_6.

□

On peut donc employer cet algorithme face à des contraintes non convexes sans craintes de cyclage mais en émettant des réserves chaque fois que l'algorithme décrétera l'incompatibilité.

§ 2. Diversification du paramètre de pénalisation

Une première tentative d'amélioration serait de considérer non plus un paramètre de pénalisation global, mais bien un paramètre particulier pour chacune des contraintes. Dès lors, la fonction de pénalisation deviendrait

$$\phi^{\star}(x,\bar{p}) = \sum_{i=1}^{m} \frac{1}{p_i} \omega(p_i \cdot c_i(x))$$

que l'on minimise sur $x \in D \subseteq R^n$ avec $\bar{p}$, vecteur des paramètres, $\bar{p} \in R^m$.

L'incrémentation des "p_i" se ferait en fonction des valeurs de la $i^{ème}$ contrainte aux minima déjà obtenus. Les résultats théoriques, à première vue, semblent pouvoir tenir face à cette modification, avec cependant certaines conditions sur l'incrémentation, mais les résultats obtenus sur ordinateur se sont avérés infructueux face à ceux de l'algorithme initial.

§ 3. Non invariance face à une prémultiplication des contraintes

Remarquons d'abord que, face aux systèmes d'inéquations

$$(P) \qquad c_i(x) \leqslant 0 \qquad i = 1, \ldots, m$$

et

$$(P') \qquad \alpha_i c_i(x) \leqslant 0 \qquad i = 1, \ldots, m$$

avec $\alpha_i > 0$, la démarche de l'algorithme est différente alors que (P) et (P') sont équivalents.

Lorsque l'on traite un problème où l'on a dès le départ une idée sur l'allure générale des contraintes, il est parfois intéressant de prémultiplier les contraintes avant de lancer l'algorithme. Cette pondération des contraintes serait effectuée en vue d'éviter des problèmes d'ordre numérique sur ordinateur.

§ 4. Traitement sur ordinateur

4.1. Overflow...

Les résultats numériques présentés par Schnabel proviennent d'un programme utilisant comme fonction pénalisante l'exponentielle, c'est-à-dire $e^y - 1$, $y \in R$.

a) Gardons l'exponentielle comme fonction pénalisante puisque c'est celle qui, intuitivement, pénalise le plus. En effet, sur les exemples présentés par Schnabel, l'algorithme donne la solution après deux ou trois incrémentations du paramètre de pénalisation. Mais dès que l'on exécute ce programme sur des exemples plus complexes, tels ceux présentés au chapitre II, on se trouve très vite face à des problèmes d'overflow qui provoquent des incohérences dans les données nécessaires à la sous-routine de minimisation. Un premier problème est le choix du paramètre de départ. En effet, initialiser celui-ci à zéro n'a pas de sens étant donné la particularité de ces contraintes. De plus, son incrémentation nécessite quelques précautions également. Voici celles utilisées dans le programme que j'ai mis au point. Notons "ωsup" la plus grande valeur réelle de y telle que

$$\omega(y) < M$$

où M est la valeur numérique maximale de l'ordinateur. Une démarche possible pour fixer p_o, paramètre initial, consiste à calculer en x_o point de départ,

$$\max_{i=1,\ldots m} \{c_i(x_o)\} = c_i^\circ$$

et prendre pour paramètre initial

$$p_o = \frac{\omega sup}{c_i^\circ} \star c$$

où c est une constante strictement plus petite que 1.

Dès lors,

$$\omega(p_o.c_i(x_o)) < M \qquad i = 1,\ldots, m$$

ce qui assure la cohérence des données initiales rentrées dans la sous-routine de minimisation.
Dans la même optique, on peut incrémenter le paramètre de pénalisation comme suit :

$$p^+ \in [\alpha p, \beta p]$$

où $0 < \alpha < \beta$. Comme précédemment, on calcule

$$\underset{i=1,\ldots m}{\text{Max}} \{c_i(x^{\star}(p))\} = c_i^{\star}$$

où $x^{\star}(p)$ est l'optimum à l'étape correspondant au paramètre p. Appelons

$$p_1 = \frac{\omega sup}{c_i^{\star}} \cdot C$$

où $c > 1$. Si $p_1 \in [\alpha p, \beta p]$, on choisit pour nouveau paramètre $p^+ = p_1$, si $p_1 < \alpha p$, on choisit $p^+ = \alpha p$ et si $p_1 > \beta p$, $p^+ = \beta p$. On utilise la borne inférieure αp pour être assuré de la convergence, car ainsi l'incrémentation du paramètre ne peut tendre vers zéro. Par contre, la signification de "β" est plus subjective. Elle dépend en fait de la sous-routine de minimisation utilisée. En effet, certaines sous-routines possèdent un mode d'exécution que l'on utilise lorsque la fonction objective ne diffère que légèrement d'un appel à l'autre.

Malgré ces précautions, l'utilisation de l'exponentielle comme fonction pénalisante provoque souvent des problèmes à moins d'avoir une sous-routine de minimisation très robuste.

b) Gardons cependant le caractère exponentiel dans la région négative mais "adoucissons la fonction pénalisante" dans la région positive par une

fonction du second degré, c'est-à-dire prenons

$$\begin{cases} \omega(y) = e^y - 1 & \text{si } y < \alpha \\ \omega(y) = \frac{1}{2}(y-\alpha)^2 e^\alpha + (y-\alpha)e^\alpha + e^\alpha - 1 & \text{sinon} \end{cases}$$

avec $\alpha > 0$.

Remarquons que le résultat théorique du § 6 au premier chapitre reste valide. De plus, on remarque expérimentalement que cette nouvelle classe de fonctions de pénalisation convient nettement mieux dès que les ensembles de contraintes deviennent plus complexes.

4.2. Gain de temps CPU

a) Il me semble intéressant de faire le test de compatibilité des contraintes à l'intérieur de la sous-routine de minimisation. Ce n'est en fait que le calcul d'un maximum de m réels mais cela peut faire arrêter plus tôt une procédure de minimisation inutile.

b) Un gain de temps majeur correspond à l'utilisation d'un test supplémentaire interne à la sous-routine de minimisation. Ce test consiste à sortir de cette sous-routine dès que la fonction objectif est négative et que le maximum des contraintes ne diminue plus assez vite. Dès lors, on augmente le paramètre et on appelle à nouveau la sous-routine, ceci afin d'atteindre beaucoup plus vite le paramètre critique. Une variante de ce test consiste à tenir compte du mode d'exécution de certaines sous-routines indiquées précédemment en exigeant un nombre de pas fixé dans chaque appel de la sous-routine.

Chapitre IV

RELAXATION DES CONTRAINTES

§ 1. Présentation du problème

Considérons le système

$$(P)\quad \begin{cases} c_i(x) \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \end{cases}$$

où c_i est non linéaire et l_j linéaire. Lorsque ce système est incompatible on a vu au premier chapitre que l'algorithme de Schnabel s'arrêtait (étape # 2 de l'algorithme II). Cependant, dans la plupart des problèmes pratiques, si l'on décerne l'incompatibilité, il semble intéressant de "relaxer" les contraintes dans le but d'obtenir un ensemble de contraintes compatible. Relaxer les contraintes reviendra à trouver des nombres positifs $\delta_1, \ldots \delta_m$ tels que le système

$$\begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \end{cases},$$

appelé système relaxé, soit compatible.

Le problème est d'effectuer cette relaxation "le mieux possible". Ceci n'est pas simple car le concept de relaxation optimale est tout à fait subjectif. Un critère parmi d'autres est de "toucher le moins possible aux contraintes". Mathématiquement, cela pourrait se traduire de la façon suivante:

Etant donnés m facteurs de pondération des contraintes non linéaires, γ_i, $\gamma_i \in]0,1[$, $i = 1, \ldots, m$, la relaxation $\delta^* = (\delta_1^*, \ldots, \delta_m^*)$ est estimée optimale si elle est solution du problème

$$\begin{cases} \underset{\substack{\delta \in R^m \\ x \in D}}{\text{Minimiser}} \quad \sum_{i=1}^{m} \gamma_i \, \delta_i \\ \\ \text{S.C.} \quad \begin{array}{ll} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m. \end{array} \end{cases}$$

Remarquons que ce type de relaxation garde les contraintes linéaires intactes. Très souvent, on désire n'effectuer la relaxation que sur les contraintes non linéaires. C'est par exemple le cas du problème présenté au chapitre II.

Le but de l'algorithme qui suit (Relaxation I) est de fournir des résultats sur une relaxation possible (et non la meilleure au sens de notre critère !) pour pouvoir tester l'efficacité de l'algorithme (Relaxation II) présenté plus loin. Ce premier algorithme n'est cependant pas dénué de tout sens intuitif.

Commençons par alléger les notations en notant par "Schnabel ((P),p,x)" l'appel de l'algorithme de Schnabel pour un système d'inéquations (P), avec pour paramètre initial "p" et comme point de départ "x". A la sortie, "p" sera la valeur finale du paramètre et "x" le point optimal trouvé dans la dernière minimisation. Notons (P_δ) le système des inéquations initiales relaxées de δ i.e.

$$(P_\delta) \quad \begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \end{cases}$$

§ 2. Algorithme Relaxation I

2.1. Algorithme

INITIALISATION Choisir p_o paramètre initial et x_o point de départ.
Choisir $\varepsilon > 0$, pasmin > 0 et $\alpha > 1$.
Poser $x = x_o$, $p = p_o$ et $\delta_i = 0$, $i = 1, \ldots, m$
et aller à l'étape principale.

ETAPE PRINCIPALE

1 Appeler la sous-routine Schnabel $((P_\delta), p, x)$, noter p et $x^\star$ les paramètres à la sortie et aller en # 2.

2 Si le système est incompatoble, aller en # 3
sinon $x^\star$ est un point admissible du système $\begin{cases} c_i(x) - \delta_i \leqslant 0, & i = 1,\ldots,m. \\ l_j(x) \leqslant 0 & j = 1,\ldots,q. \end{cases}$
STOP

3 Noter i_o l'indice de $I = \{1,2,\ldots,m\}$ tel que

$$c_{i_o}(x^\star) - \delta_{i_o} = \max_{k \in I} [c_k(x^\star) - \delta_k]$$

et noter j_o l'indice de $I \setminus \{i_o\}$ tel que

$$c_{j_o}(x^\star) - \delta_{j_o} = \max_{k \in I \setminus \{i_o\}} [c_k(x^\star) - \delta_k]$$

Calculer

$$\delta^+_{i_o} = \text{Max}\ \{(c^\star_{i_o} - c^\star_{j_o}).\ \alpha,\ \text{pasmin}\}$$

où $c^\star_{i_o} = c_{i_o}(x^\star) - \delta_{i_o}$ et $c^\star_{j_o} = c_{j_o}(x^\star) - \delta_{j_o}$

Poser $\delta_{i_o} = \delta_{i_o} + \delta^+_{i_o}$ et aller en # 1.

2.2. Explications et justification intuitive

Cet algorithme consiste à relaxer une seule contrainte chaque fois que l'on détecte l'incompatibilité. Le choix de cette contrainte est justifié plus loin. Elle est en fait celle qui est la plus violée au point $x^{\star}$ sorti par la sous-routine de Schnabel.

Si l'on considère notre critère d'optimalité d'une relaxation avec pour facteurs de pondération $\gamma_i = 1$ pour $i = 1, \ldots, m$, on peut justifier intuitivement la démarche de l'algorithme. A chaque itération, pour tenter de vérifier ce critère, on va allouer une relaxation "$\Delta\delta$" supplémentaire à la contrainte qui va provoquer une diminution maximale de la violation des contraintes et cela, bien sûr, "en moyenne", c'est-à-dire que, pour une même relaxation "$\Delta\delta$", le but va être de chercher quelle est la contrainte qui, en la relaxant de "$\Delta\delta$", provoquera une diminution maximale de cette "moyenne" des contraintes. Mais très vague est cette notion de moyenne. Cependant une diminution d'une contrainte déjà satisfaite est superflue; par contre, plus une contrainte est violée, plus il semble intéressant de la pondérer fortement dans cette moyenne. Remarquons qu'à la sortie de l'algorithme de Schnabel, nous disposons justement d'une moyenne des contraintes "c_i" satisfaisant notre intuition: c'est la fonction objective $\phi(x,p)$ que l'on notera dorénavant $\phi(x,p,\delta)$ bien que les "δ_i" n'apparaissent pas comme variables dans la sous-routine de Schnabel mais bien comme paramètres que l'on ajuste en dehors de celle-ci.

Cherchant à minimiser $\sum_{i=1}^{m} \delta_i$, on va allouer une relaxation "$\Delta\delta$" à la contrainte qui va, au premier ordre, infliger une diminution maximale de la fonction objective. Pour cela, calculons les dérivées au point $x^{\star}$.

$$\frac{\partial \phi(x^{\star},\delta,p)}{\partial \delta_i} \qquad\qquad i = 1, \ldots, m$$

et considérons la plus négative. On a immédiatement

$$\frac{\partial \phi\ (x^{\star},\ \delta,\ p)}{\partial\ \delta_i} = \frac{\partial}{\partial \delta_i} \left[\frac{1}{p} \sum_{i=1}^{m} \omega(p.(c_i(x^{\star}) - \delta_i)) \right]$$

$$= - \omega'(p.(c_i(x^{\star}) - \delta_i)) \qquad i = 1, \ldots, m.$$

Comme ω' est positive et croissante en vertu de la définition d'une fonction pénalisante (ii) et (iii), on a que l'expression numérique

$$\frac{\partial\ \phi(x^{\star},\ \delta,\ p)}{\partial\ \delta_i}$$

est la plus négative pour i_o tel que

$$c_{i_o}(x^{\star}) - \delta_{i_o} = \max_{k \in I} (c_k(x^{\star}) - \delta_k)$$

A chaque sortie de la sous-routine de Schnabel, il semble donc intéressant d'accorder une relaxation supplémentaire à la contrainte de (P_δ) la plus violée en $x^{\star}$.

Le problème est alors de choisir "$\Delta\delta$" car les justifications précédentes ne sont valables qu'au premier ordre, donc pour "$\Delta\delta$" = $\delta^{+}_{i_o}$ très petit.

Remarque 1: Une contrainte que l'on vient de relaxer ne va pas diminuer dans la minimisation suivante.

Preuve: Pour montrer cela, introduisons les notations suivantes et rappelons que $x^{\star}$ est un minimum de $\phi(x, \delta, p)$. Notons $c_i(x^{\star}) = c_i$, $i = 1, \ldots, m$ et i_o, l'indice tel que

$$c_{i_o} - \delta_{i_o} = \max_{k \in I} (c_k - \delta_k)$$

$\hat{c}_{i_o}(x) - \delta_{i_o}$ va donc être relaxée et devenir

$$c_{i_o}(x) - \delta'_{i_o} = c_{i_o} - (\delta_{i_o} + \delta^+_{i_o}).$$

Soit $\delta = \begin{pmatrix} \delta_1 \\ \vdots \\ \delta_{i_o} \\ \vdots \\ \delta_m \end{pmatrix}$ et notons $\delta' = \begin{pmatrix} \delta_1 \\ \vdots \\ \delta'_{i_o} \\ \vdots \\ \delta_m \end{pmatrix}$ c'est-à-dire seule la

i_o composante δ'_{i_o} diffère : $\delta'_{i_o} > \delta_{i_o}$

Considérons un déplacement Δx tel que $x^\star + \Delta x$ soit encore admissible par rapport aux contraintes linéaires, $l_j(x) \leqslant 0$, $j = 1, \ldots, q$, et notons

$$\Delta c_j = c_j(x^\star) - c_j(x^\star + \Delta x) \qquad j = 1, \ldots, m$$

ou encore

$$c_j(x^\star + \Delta x) = c_j - \Delta c_j \qquad j = 1, \ldots, m$$

Ecrivons la fonction objective en séparant le terme intéressant: au point $x^\star$, on a

$$\phi(x^\star, \delta, p) = \frac{1}{p} \sum_{\substack{j=1 \\ j \neq i_o}}^{m} \omega(p.(c_j - \delta_j)) + \frac{1}{p}\, \omega(p.(c_{i_o} - \delta_{i_o}))$$

et en $x^\star + \Delta x$

$$\phi(x^\star + \Delta x, \delta, p) = \frac{1}{p} \sum_{\substack{j=1 \\ j \neq i_o}}^{m} \omega(p.(c_j - \delta_j)) + \Delta A + \frac{1}{p}\, \omega(p.(c_{i_o} - \delta_{i_o})) + \Delta B$$

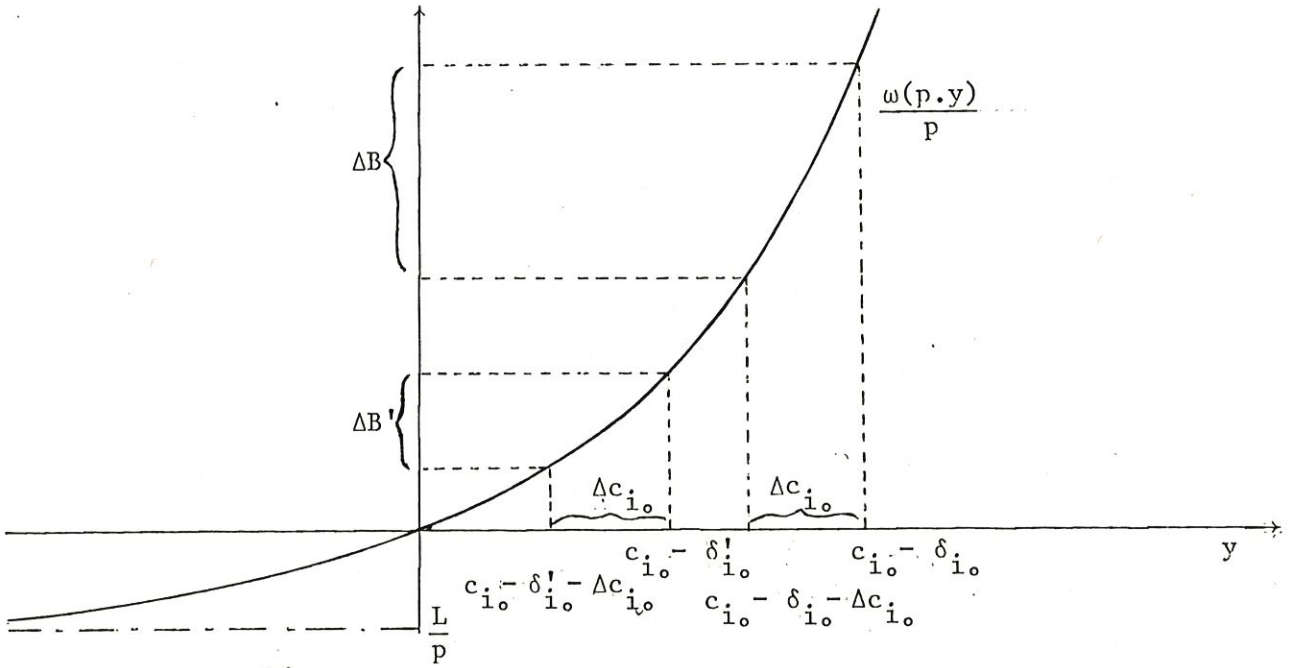
où $$\Delta A = \frac{1}{p} \sum_{\substack{j=1 \\ j \neq i_o}}^{m} \omega(p.(c_j - \Delta c_j - \delta_j)) - \frac{1}{p} \sum_{\substack{j=1 \\ j \neq i_o}}^{m} \omega(p.(c_j - \delta_j))$$

et $$\Delta B = \frac{1}{p} \omega(p.(c_{i_o} - \Delta c_{i_o} - \delta_{i_o})) - \frac{1}{p} \omega(p.(c_{i_o} - \delta_{i_o}))$$

or $x^{\star}$ est un minimum de $\phi(., \delta, p)$ et donc

$$\Delta A \geqslant - \Delta B \qquad (4.1)$$

Considérons les mêmes équations pour la nouvelle fonction objectif $\phi(., \delta', p)$ aux points $x^{\star}$ et $x^{\star} + \Delta x$ et prenons des notations similaires $\Delta A'$ et $\Delta B'$. Le même déplacement Δx provoquera pour la nouvelle fonction objectif $\phi(., \delta', p)$ un "$\Delta A'$" égal à "ΔA" (car les (m-1) termes de la somme sont inchangés) mais par contre, un $\Delta B'$ tel que $|\Delta B'| < |\Delta B|$. En effet, voyons le graphe du terme en question où l'on considère un déplacement Δx provoquant une diminution de la $i_o^{\text{ième}}$ contrainte (analogue dans l'autre cas)



on a bien

$$|\Delta B'| < |\Delta B| \qquad (4.2)$$

Pour démontrer la remarque 1, montrons que pour tout x du domaine $\mathbb{D}$ vérifiant les contraintes linéaires et tel que $c_{i_o}(x) < c_{i_o}(x^{\star})$, nous avons $\phi(x, \delta', p) > \phi(x^{\star}, \delta', p)$. Notons $x = x^{\star} + \Delta x$ et $c_{i_o}(x^{\star} + \Delta x) = c_{i_o} - \Delta c_{i_o}$ où $\Delta c_{i_o} > 0$. Pour ce déplacement Δx, on a successivement par (4.1) et (4.2)

$$\Delta B \leqslant 0$$

$$\Delta A \geqslant - \Delta B \geqslant 0$$

$$\Delta A' = \Delta A \geqslant - \Delta B > - \Delta B'$$

et donc $\Delta A' + \Delta B' > 0$ i.e. $\phi(x, \delta', p) > \phi(x^{\star}, \delta', p)$.

□

Remarque 2: Diminution "en moyenne" des contraintes non relaxées durant la minimisation suivante.

Preuve: Gardons les notations de la démonstration précédente et appelons $x^{\star\star}$ la solution du nouveau problème de minimisation

$$\begin{cases} \min\limits_{x \in D} \ \phi(x, \delta', p) \\ \\ \quad \text{sc} \ \ l_j(x) \leqslant 0 \qquad\qquad j = 1, \ldots, q \\ \qquad\ \ \delta_i \geqslant 0 \qquad\qquad i = 1, \ldots, m \end{cases}$$

En vertu de la remarque précédente, $x^{\star\star} = x^{\star} + \Delta x$ est tel que $c_{i_o}(x^{\star\star}) \geqslant c_{i_o}(x^{\star})$ et donc $\Delta B'$ est positif ou nul. Or

$$\phi(x^{\star\star}, \delta', p) \leqslant \phi(x^{\star}, \delta', p)$$

puisque $x^{\star\star}$ est minimum de $\phi(., \delta', p)$ et donc $\Delta A' + \Delta B' \leqslant 0$, c'est-à-dire

$$\Delta A' \leqslant - \Delta B' \leqslant 0$$

ce qui traduit une diminution en "moyenne" des autres contraintes $(j \neq i_o)$.

□

Ces deux résultats nous montrent que si l'on prend $\delta^+_{i_o} = \Delta\delta$ trop petit, après une nouvelle minimisation, $(c_{i_o}(x^{\star\star}) - \delta^!_{i_o})$ sera encore la contrainte la plus violée et on la relaxera à nouveau. On peut donc d'emblée choisir $\delta^+_{i_o}$ plus grand. Les résultats précédemment démontrés m'ont poussé à choisir

$$\delta^+_{i_o} = \text{Max } [\{(c_{i_o} - \delta_{i_o}) - (c_{j_o} - \delta_{j_o})\} \cdot \alpha \ , \text{pasmin}]$$

où i_o et j_o sont définis comme dans l'algorithme et α est une constante à déterminer, $\alpha > 1$. L'utilisation de la constante positive "pasmin" a pour seul but d'assurer la convergence de l'algorithme puisque l'on relaxera à chaque itération d'une quantité supérieure à cette constante.

2.3. Conclusion

Ce premier algorithme a, bien entendu, de gros inconvénients, le premier d'entre eux étant l'absence totale de résultats théoriques quant à l'optimalité de la relaxation. Mais de plus, on ne relaxe jamais qu'une seule contrainte à chaque itération, ce qui provoque une hausse trop grande du temps d'exécution dès l'utilisation de l'algorithme face à un problème avec plus de contraintes.

Par contre, un avantage considérable me semble être le fait que, dans chaque appel de la sous-routine de minimisation, le nombre des variables est et reste le nombre des variables initiales. En effet, les "δ_i" ne sont modifiés qu'en dehors de la sous-routine. De plus, à chaque appel de la sous-routine de Schnabel, on réutilise toute l'information précédente: le paramètre de pénalisation critique ainsi que le dernier point trouvé. Mais cet algorithme est basé sur l'intuition seule et a pour unique but la comparaison avec l'algorithme Relaxation II qui suit.

§ 3. Algorithme de Relaxation II

3.1. Motivation et démarche intuitive

Rappelons que notre problème est un problème de minimisation à (n+m) variables avec m contraintes non linéaires de la forme:

$$\begin{cases} \underset{\substack{\delta \in R^m \\ x \in D}}{\text{Min}} \ \sum_{i=1}^{m} \gamma_i \, \delta_i \\ \text{s.c.} \quad c_i(x) - \delta_i \leqslant 0 \qquad i = 1, \ldots, m \\ \qquad\quad l_j(x) \leqslant 0 \qquad\quad j = 1, \ldots, q \\ \qquad\quad \delta_i \geqslant 0 \qquad\quad i = 1, \ldots, m \end{cases} \tag{4.3}$$

où les $l_j(x)$, $j = 1, \ldots, q$, sont linéaires.

Comme ce problème peut être très coûteux à résoudre directement, une démarche possible est d'essayer de ramener sa résolution à une suite de problèmes de minimisations sans contraintes non linéaires. Ce sont les méthodes dites de pénalisation.

Dans cet ordre d'idée, nous allons essayer d'adapter la méthode de Schnabel à notre cas, c'est-à-dire travailler par pénalisation pour éliminer les contraintes non linéaires en les pénalisant dans la fonction objectif.

Plus précisément, la résolution du problème (4.3) va être remplacée par l'application itérative de l'algorithme de Schnabel à des systèmes d'inéquations que l'on modifie légèrement à chaque itération.

Le premier d'entre eux, (P')

$$(P') \begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \\ \sum_{i=1}^{m} \gamma_i \delta_i \leqslant 0 \end{cases}$$

est équivalent au système initial (P)

$$(P) \begin{cases} c_i(x) \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \end{cases}$$

La dernière contrainte de (P'), bien que linéaire, sera traitée comme non linéaire dans l'algorithme de Schnabel en ce sens qu'elle apparaîtra dans la fonction objectif $\phi(x,\delta,p)$. Dès lors, le problème de minimisation de la méthode de Schnabel appliquée à (P') s'écrit :

$$(4.4) \begin{cases} \underset{\substack{x \in D \\ \delta \in R^m}}{\text{Min}} \ \phi(x,\delta,p) = \frac{1}{p} \sum_{i=1}^{m} \omega(p.(c_i(x) - \delta_i)) + \frac{1}{p} \omega(p \left[\sum_{i=1}^{m} \gamma_i \delta_i \right]) \\ \text{s.c.} \quad l_j(x) \leqslant 0 \qquad j = 1, \ldots, q \\ \qquad\quad \delta_i \geqslant 0 \qquad i = 1, \ldots, m \end{cases}$$

Ayant appliqué l'algorithme de Schnabel au système initial (P) (supposé incompatible), on obtient l'information suivante :

1. Un paramètre de pénalisation critique qui a décrété l'incompatibilité de (P) et
2. un point $x^{\star}(p)$ qui va servir comme point de départ en initialisant δ_i à zéro, i = 1, ..., m.

C'est pourquoi, appliquant par après l'algorithme de Schnabel à (P'), on détecte très rapidement son caractère incompatible. Notons par $(x^{\star}(p), \delta(p))$ le dernier minimum de (4.4) trouvé. Soit "R" la valeur optimale du problème

(4.3), c'est-à-dire la valeur de la fonction objectif du problème (4.3) correspondant à la meilleure relaxation au sens de notre critère.

Posons, si $(c_i(x^*(p)) - \delta_i^*(p)) \leqslant 0$

$$\delta'_i = \delta_i^*(p)$$

et, dans le cas contraire

$$\delta'_i = \delta_i^*(p) + c_i(x^*(p)) - \delta_i^*(p) = c_i(x^*(p))$$

Notons $Bsupr = \sum_{i=1}^{m} \gamma_i \delta'_i$. Bsupr est une borne supérieure pour la relaxation. En effet, on est assuré que $R \leqslant Bsupr$ car, en vertu de l'expression des δ'_i, $x^*(p)$ vérifie

$$\begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \\ \sum_{i=1}^{m} \gamma_i \delta_i = Bsupr \end{cases}$$

en $\delta_i = \delta'_i$

De plus, on est assuré de l'incompatibilité de (P) et a fortiori de (P'). Par conséquent, R est strictement positif. Initialisant Binfr à zéro, on obtient

$$Binfr < R \leqslant Bsupr \qquad (4.7)$$

Il s'agit ensuite d'appliquer à nouveau l'algorithme de Schnabel mais sur un problème légèrement modifié. Une première idée serait d'utiliser une méthode de dichotomie, c'est-à-dire appliquer l'algorithme de Schnabel au système (P")

$$(P'') \quad \begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1; \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \\ \sum_i^m \gamma_i \delta_i \leqslant \dfrac{Bsupr - Binfr}{2} \end{cases}$$

Si (P") est compatible poser

$$Bsupr \longleftarrow \frac{Bsupr - Binfr}{2}$$

et dans le cas contraire

$$Binfr \longleftarrow \frac{Bsupr - Binfr}{2}$$

et ainsi de suite...

Mais cette méthode est fort primaire et très coûteuse si l'on désire une précision suffisante.
De plus, après avoir appliqué l'algorithme de Schnabel à (P'), il n'est pas intéressant d'initialiser Binfr à zéro: on peut le vérifier expérimentalement, R se trouve souvent très près de Bsupr.

En ce moment se posent les problèmes suivants :

1) Comment estimer Binfr plus valablement dès le départ (i.e. après avoir appliqué l'algorithme de Schnabel à (P').

2) Comment obtenir à chaque itération de l'information sur Binfr et Bsupr en même temps.

Pour répondre à ces questions, utilisons un indice, "k", pour différencier chaque itération. Initialisons-le :

$$k = 1$$

et posons

$$Binfr(k) = 0.$$

Au lieu de procéder comme avant, considérons la suite des systèmes

$$(P_k)_k \quad \begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \\ \sum_{i=1}^{m} \gamma_i \delta_i \leqslant \text{Binfr}(k) \end{cases}$$

auxquels nous appliquons itérativement l'algorithme de Schnabel avec comme paramètre de pénalisation le dernier obtenu. Le problème est de mettre à jour Binfr(k) de sorte que

$$\text{Binfr}(k) \leqslant \text{Binfr}(k+1) < R$$

pour tout k, i.e. à chaque itération.

Remarquons qu'en utilisant Binfr(k) dans la dernière inéquation du système (P_k), nous savons dès le départ que (P_k) est strictement incompatible. Remarquons de plus qu'en modifiant (P_k) en (P_{k+1}) par une incrémentation de Binfr(.), les "δ_i" vont en moyenne augmenter dans la(les) minimisation(s) interne(s) à l'algorithme de Schnabel appliqué à (P_{k+1}). Donc, d'une itération à l'autre, les m premières inéquations vont avoir tendance à être violées de moins en moins et parallèlement, le dernier terme de la fonction objectif de l'algorithme de Schnabel reflétera de plus en plus la minimisation de "$\sum_{i=1}^{m} \gamma_i \delta_i$".

3.2. Algorithme (Relaxation II)

INITIALISATION Choisir p_o paramètre initial, $x_o \in D$ point de départ
Choisir ε_1, $\varepsilon_2 > 0$ et $\beta \geqslant 1$.
Poser $p = p_o$, $x = x_o$ et aller à l'étape principale

ETAPE PRINCIPALE

(# 1) # a Résoudre le problème

$$\begin{cases} \underset{x \in D}{\text{Min}}\ \phi(x,p) \triangleq \dfrac{1}{p} \sum_{i=1}^{m} \omega(p.c_i(x)) \\ \text{s.c. } 1_j(x) \leqslant 0 \qquad j = 1, \ldots, q \end{cases}$$

Noter $\phi^{\star}(p)$ la valeur optimale du problème et $x^{\star}(p)$ le minimum.

b Si $\phi^{\star}(p) > 0$ aller en (#2)
sinon aller en # c

c Si $\underset{1 \leqslant i \leqslant m}{\text{Max}}\ c_i(x^{\star}(p)) \leqslant \varepsilon_1$ aller en (#3)
sinon $p = p + p^{+}$ et aller en # a.

(# 2) Incompatibilité du système $(P) \equiv \begin{cases} c_i(x) \leqslant 0 & i = 1, \ldots, m \\ 1_j(x) \leqslant 0 & j = 1, \ldots, q \end{cases}$

Si on veut utiliser la procédure de relaxation aller en (# 4)
sinon STOP

(# 3) Comptabilité en $x^{\star}(p)$ STOP

(# 4) Choisir γ_i facteur de pondération de la $i^{ème}$ contrainte tel que

$$0 < \gamma_i < 1 \qquad i = 1, \ldots, m$$

Poser $k = 1$

$Binfr(k) = 0$

$Bsupr(k) = 10.D30$

$\delta_i = 0 \qquad i = 1, \ldots, m$

et aller en (# 5)

(# 5) # a Résoudre le problème

$$\begin{cases} \text{Min } \phi(x, \delta, p) \triangleq \dfrac{1}{p} \displaystyle\sum_{i=1}^{m+1} \omega(p.c_i^r(x, \delta)) \\ \text{s.c. } l_j(x, \delta) = l_j(x) \leqslant 0 \qquad j = 1, \ldots, q \\ \delta_i \geqslant 0 \qquad i = 1, \ldots, m \end{cases}$$

où $c_i^r(x, \delta) = c_i(x) - \delta_i \qquad i = 1, \ldots, m$

$$c_{m+1}^r(x, \delta) = \frac{\displaystyle\sum_{i=1}^{m} \gamma_i \delta_i - Binfr(k)}{\beta}$$

Noter $\phi^{\star}(p)$ la valeur optimale du problème et $(x^{\star}(p), \delta^{\star}(p))$ le minimum.

b Si $\phi^{\star}(p) > 0$ aller en (# 6)
sinon $p = p + p^{+}$ et aller en # a.

(# 6) Calculer

$$D_k \triangleq \phi^{\star}(p) - \frac{1}{p} \omega(p.c_{m+1}^r(x^{\star}(p), \delta^{\star}(p)))$$

Si $D_k > 0$ aller en (# 7)

Sinon poser $p = p + p^+$

$Binfr(k+1) = Binfr(k)$

$k = k+1$

et aller en (# 5).

(# 7) Poser

$$Binfr(k+1) = Binfr(k) + \beta c^r_{m+1}(x^\star(p),\ \delta^\star(p))$$

Soit

$$\delta^k_i = \delta^\star_i(p) + \begin{cases} 0 & \text{si } c^r_i(x^\star(p),\ \delta^\star(p)) \leqslant 0 \\ c^r_i(x^\star(p), \delta^\star(p)) & \text{dans le cas contraire} \end{cases}$$

Poser

$$Bsupr(k+1) = \text{Min}\ (Bsupr(k),\ \sum_{i=1}^{m} \gamma_i\ \delta^k_i)$$

Si $[Bsupr(k+1) - Binfr(k+1)] < \varepsilon_2$ STOP

Sinon poser $k = k+1$ et aller en (# 5).

Remarques: 1) l'utilisation de "β" dans la dernière inéquation de chaque système (P_k) ne change rien à celle-ci puisque β est strictement positif. On montrera plus loin son utilité.

2) Pour le bon fonctionnement de cet algorithme, il y a trois résultats à démontrer; c'est l'objet du paragraphe suivant.

§ 4. Résultats théoriques de l'algorithme (Relaxation II)

4.1. Résultats à obtenir

1. La mise à jour de Binfr(k) (voir (# 6,7)) est correcte. Ce qui revient à dire, avec les notations précédentes, que

$$\text{Binfr}(K+1) < R$$

ou encore, en vertu de la définition de R, que le système

$$(P_{k+1}) \begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \\ \sum_{i=1}^{m} \gamma_i \delta_i \leqslant \text{Binfr}(k+1) \end{cases}$$

est strictement incompatible.

2. Lorsque "D_k" est négatif, (D_k est défini dans l'algorithme), on n'a aucune information sur Binfr(k+1) et, dès lors, on relance l'algorithme de Schnabel sur le même système ($(P_{k+1}) \equiv (P_k)$) mais après avoir incrémenté le paramètre de pénalisation. Il faut cependant s'assurer qu'en augmentant ce paramètre p, D_k deviendra positif.

3. Remarques quant à la convergence.

4.2. Notations de l'algorithme

Soit (P_k) le système d'inéquations

$$(P_k) \begin{cases} c_i(x) - \delta_i \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \\ \delta_i \geqslant 0 & i = 1, \ldots, m \\ \dfrac{\sum_{i=1}^{m} \gamma_i \delta_i - \text{Binfr}(k)}{\beta} \leqslant 0 \end{cases}$$

Comme l'indice k est incrémenté d'une unité à chaque entrée dans (# 5), on notera p_k le paramètre critique pour le problème (P_k), c'est-à-dire le paramètre de pénalisation qui détecte l'incompatibilité de (P_k). "p_k" est donc le paramètre sortant de l'étape (# 5). Dans la suite de ce chapitre, on utilisera aussi les notations suivantes :

★ $\phi(x, \delta, p_k)$: fonction objectif de l'algorithme de Schnabel avec pour paramètre de pénalisation, p_k, paramètre critique du système (P_k). C'est donc la fonction objectif minimisée juste avant la "k"$^{\text{ième}}$ sortie de l'étape (# 5).

★ $\phi^{\star}(p_k)$: sa valeur optimale

★ $(x^{\star}(p_k), \delta^{\star}(p_k))$: un minimum de $\phi(x, \delta, p_k)$

★ $c^r_{i,k}(x,\delta) = c_i(x) - \delta_i \qquad i = 1, \ldots, m$

$$c^r_{m+1,k}(x,\delta) = \frac{\sum_{i=1}^{m} \gamma_i \delta_i - \text{Binfr}(k)}{\beta}$$

★ $$D_k = \phi^{\star}(p_k) - \frac{1}{p_k}\omega[\, p_k \cdot c^r_{m+1,k}(x^{\star}(p_k), \delta^{\star}(p_k))]$$

ou encore

$$D_k = \frac{1}{p_k} \sum_{i=1}^{m} \omega[\, p_k \cdot c^r_{i,k}(x^{\star}(p_k), \delta^{\star}(p_k))]$$

Voyons maintenant qu'à chaque itération, le système (P_k) est strictement incompatible.

4.3. Résultat 1

La mise à jour de Binfr(k) (voir (# 6,7)) est correcte, c'est-à-dire Binfr(k) < R pour tout k.

Démonstration: Etant donné la remarque faite au paragraphe 4.1, il suffit de montrer par récurrence que (P_k) est strictement incompatible pour tout k.

a) Pour k = 1, il est immédiat que (P_1) est strictement incompatible. ((P_1) est équivalent au système (P)).

b) Supposons par récurrence que Binfr(k) < R, c'est-à-dire que (P_k) est strictement incompatible ou encore que $\phi^{\star}(p_k) > 0$. Montrons que Binfr(k+1) < R.

De deux choses l'une. Si $D_k \leqslant 0$, alors Binfr(k+1) = Binfr(k) ((# 6)) et donc Binfr(k+1) < R. Si $D_k > 0$, alors on a ((# 7)) :

$$\text{Binfr}(k+1) = \text{Binfr}(k) + \beta\, c^{r}_{m+1,k}(x^{\star}(p_k), \delta^{\star}(p_k)).$$

Supposons par l'absurde que Binfr(k+1) $\geqslant$ R. Par définition de "R", il existe un point $(x^{\star\star}, \delta^{\star\star})$ de D x R^m vérifiant les contraintes linéaires et tel que

$$\begin{cases} c_i(x^{\star\star}) - \delta_i^{\star\star} \leqslant 0 & i = 1, \ldots, m \\ \sum_{i=1}^{m} \gamma_i\, \delta_i^{\star\star} \leqslant R \leqslant \text{Binfr}(k+1) & \end{cases} \tag{4.8}$$

Par les inégalités (4.8) et par la définition d'une fonction pénalisante, nous avons alors l'inégalité suivante:

$$\phi(x^{\star\star}, \delta^{\star\star}, p_k) \leqslant \frac{1}{p_k}\, \omega(p_k . c^{r}_{m+1,k}(x^{\star\star}, \delta^{\star\star})) \tag{4.9}$$

D'autre part

$$c^{r}_{m+1,k}(x^{\star\star}, \delta^{\star\star}) \triangleq \frac{\sum_{i=1}^{m} \gamma_i\, \delta_i^{\star\star} - \text{Binfr}(k)}{\beta}$$

Majorant, dans cette égalité, $\sum_{i=1}^{m} \gamma_i\, \delta_i^{\star\star}$ par Binfr(k+1) en vertu de (4.8), nous obtenons

$$c^{r}_{m+1,k}(x^{\star\star}, \delta^{\star\star}) \leqslant \frac{\text{Binfr}(k+1) - \text{Binfr}(k)}{\beta}$$

et donc, par définition de Binfr(k+1),

$$c^r_{m+1,k}(x^{\star\star}, \delta^{\star\star}) \leqslant \frac{\not{\phi}\ c^r_{m+1,k}(x^{\star}(p_k),\ \delta^{\star}(p_k))}{\not{\phi}} \qquad (4.10)$$

Par ailleurs, la définition de D_k nous donne :

$$\phi^{\star}(p_k) = D_k + \frac{1}{p_k}\ \omega\ [\, p_k \cdot\ c^r_{m+1,k}(x^{\star}(p_k),\ \delta^{\star}(p_k))]$$

et D_k étant strictement positif,

$$\phi^{\star}(p_k) > \frac{1}{p_k}\ \omega\ [\, p_k . c^r_{m+1,k}(x^{\star}(p_k),\ \delta^{\star}(p_k))]$$

Or, ω est strictement croissante et donc (4.10) et (4.9) nous donnent successivement les inégalités suivantes :

$$\phi^{\star}(p_k) > \frac{1}{p_k}\ \ \omega(p_k \cdot\ c^r_{m+1,k}(x^{\star\star}, \delta^{\star\star}))$$

$$\geqslant \phi(x^{\star\star},\ \delta^{\star\star},\ p_k)$$

ce qui ne se peut en vertu de la définition de $\phi^{\star}(p_k)$. Notre hypothèse est absurde et donc Binfr(k+1) < R.

□

4.4. Conséquence du Résultat 1

$(\text{Binfr}(k))_k$ est une suite croissante (4.11)

$(\text{Bsupr}(k))_k$ est une suite décroissante (4.12)

Démonstration: (4.12) est évident, il suffit de regarder la mise à jour de Bsupr(k) ((# 7)).

Démontrons (4.11), i.e. Binfr(k+1) $\geqslant$ Binfr(k) pour tout k.
Si $D_k \leqslant 0$, c'est évident. Considérons le cas où $D_k \succeq 0$. Remarquons d'abord qu'en vertu du résultat 1, (P_k) est strictement incompatible et donc, en $(x^{\star}(p_k), \delta^{\star}(p_k))$, il existe i_o tel que

$$c^r_{i_o,k}(x^{\star}(p_k), \delta^{\star}(p_k)) > 0$$

Si $i_o = m+1$, on a la thèse puisque

$$\text{Binfr}(k+1) = \text{Binfr}(k) + \beta\, c^r_{m+1,k}(x^{\star}(p_k), \delta^{\star}(p_k))$$

où β est une constante strictement positive. Sinon, on a, par le caractère optimal du point $(x^{\star}(p_k), \delta^{\star}(p_k))$

$$\frac{\partial}{\partial \delta_{i_o}} \phi(x^{\star}(p_k), \delta^{\star}(p_k), p_k) \geqslant 0$$

ou encore

$$(-1).\omega'[p_k.c^r_{i_o,k}(x^{\star}(p_k),\delta^{\star}(p_k))] + \frac{\gamma_{i_o}}{\beta}.\left[\omega'[p_k.c^r_{m+1,k}(x^{\star}(p_k),\delta^{\star}(p_k))]\right] \geqslant 0$$

or $\quad 0 \leqslant \dfrac{\gamma_{i_o}}{\beta} < 1 \quad$ et donc

$$\omega'[p_k\ c^r_{i_o,k}(x^{\star}(p_k),\delta^{\star}(p_k))] < \omega'[p_k.c^r_{m+1,k}(x^{\star}(p_k), \delta^{\star}(p_k))]$$

ω' étant positive et croissante par définition d'une fonction pénalisante, on obtient

$$0 < c^r_{i_o,k}(x^{\star}(p_k),\delta^{\star}(p_k)) < c^r_{m+1,k}(x^{\star}(p_k),\delta^{\star}(p_k))$$

et donc Binfr(k+1) > Binfr(k) $\qquad \square$

4.5. Résultat 2

Plaçons-nous à une itération quelconque, "k_o", et montrons que $(D_k)_{k \leqslant k_o}$ ne restera pas négatif ou nul à partir de cette itération (car s'il le restait, on ne remettrait plus à jour Binfr(.) et dès lors l'algorithme bouclerait).

Démonstration: Par l'absurde, supposons qu'à partir de l'itération k_o, i.e. pour $k \geqslant k_o$, D_k reste négatif ou nul. L'algorithme dans ce cas incrémente le paramètre de pénalisation p. Supposons donc que, pour tout $k \geqslant k_o$

$$D_k \leqslant 0 \tag{4.15}$$

a) Montrons d'abord que

$\forall\ \varepsilon_1 > 0\ \ \exists\ p_1$ tel que $\forall\ k \geqslant k_o$ tel que $p_k \geqslant p_1$, on a

$$\left[\underset{1 \leqslant i \leqslant m}{\text{Max}} \ [c^r_{i,k}(x^\star(p_k), \delta^\star(p_k))] \right] \leqslant \varepsilon_1 \tag{4.16}$$

Remarquons que l'on prend le minimum sur les indices correspondant aux inéquations du problème initial (P). Pour démontrer (4.16), choisissons p_1 comme suit: soit y_1 tel que $\omega(y_1) > -(m-1)L$ et posons $p_1 = \frac{y_1}{\varepsilon_1}$. Supposons par l'absurde que, pour $k \geqslant k_o$ tel que $p_k \geqslant p_1$, on ait

$$\underset{1 \leqslant i \leqslant m}{\text{Max}} -\left[c^r_{i,k} \ [x^\star(p_k), \delta^\star(p_k)] \right] > \varepsilon_1 . \tag{4.13}$$

Or, par définition,

$$D_k = \frac{1}{p_k} \sum_{i=1}^{m} \omega[p_k \cdot c^r_{i,k}(x^\star(p_k), \delta^\star(p_k))] .$$

Dès lors, par (4.13) et les propriétés d'une fonction pénalisante, entre autres [ch. I, (ii) et (iv)], nous avons

$$D_k > \frac{1}{p_k} (m-1)L + \frac{\omega(p_k \cdot \varepsilon_1)}{p_k}$$

D'autre part, $p_k \cdot \varepsilon_1 \geqslant p_1 \varepsilon_1 = y_1$ et donc utilisant à nouveau [ch. I, (ii)], on obtient la minoration :

$$D_k > \frac{1}{p_k}(m-1)L + \frac{1}{p_k}[-(m-1)L] = 0$$

ce qui contredit l'hypothèse (4.15).

b) Voyons qu'en vertu de (4.16), nous avons

$\forall\, \varepsilon_2 > 0$, $\exists\, p_2$ tel que $\forall\, k \geqslant k_o$ tel que $p_k \geqslant p_2$, on a

$$c^r_{m+1,k}(x^\star(p_k),\ \delta^\star(p_k)) < \varepsilon_2 \qquad (4.14)$$

Nous nous limiterons à faire la démonstration pour la classe de fonction de pénalisation suivante:

$$\omega(y) = \begin{cases} e^y - 1 & y \leqslant \alpha \\ \frac{1}{2}(y-\alpha)^2 e^\alpha + (y-\alpha)e^\alpha + e^\alpha - 1 & \text{sinon} \end{cases}$$

avec $\alpha \geqslant 1$.

Notons $\gamma \triangleq \min_{1 \leqslant i \leqslant m} \gamma_i$ et considérons p_3 tel que

$$p_3 > \frac{\alpha . 2\ \beta}{\gamma \cdot \varepsilon_2} \qquad (4.17)$$

Soit de plus p_4 tel que $\forall\, k \geqslant k_o$ tel que $p_k \geqslant p_4$

$$\left[\max_{1 \leqslant i \leqslant m} [\, c^r_{i,k}[x^\star(p_k),\ \delta^\star(p_k)]\right] \leqslant \frac{\gamma \varepsilon_2}{2\ \beta} = \varepsilon_3 \qquad (4.18)$$

ce qui est possible en vertu de (4.16) démontré en a). Choisissons p_2 comme suit :

$$p_2 = \text{Max}\ (p_3, p_4) \qquad (4.19)$$

et supposons par l'absurde qu'il existe $k \geqslant k_o$ tel que $p_k \geqslant p_2$ et

$$c^r_{m+1,k}(x^\star(p_k),\ \delta^\star(p_k)) > \varepsilon_2 \tag{4.20}$$

il existe donc i_o tel que $\delta^\star_{i_o}(p_k) > 0$. Pour cet indice, calculons

$$\frac{\partial \phi}{\partial \delta_{i_o}}[x^\star(p_k), \delta^\star(p_k), p_k] = (-1)\omega'\left[p_k . c^r_{i_o,k}(x^\star(p_k), \delta^\star(p_k))\right]$$

$$+ \frac{\gamma_{i_o}}{\beta} \cdot \omega'\left[p_k . c^r_{m+1,k}[x^\star(p_k), \delta^\star(p_k)]\right]$$

Or ω' est positive et croissante en vertu de [ch. I, (ii), (iii)]. Comme $p_k \geqslant p_2$, on a, utilisant (4.17, 4.18, 4.19), la minoration

$$\frac{\partial \phi}{\partial \delta_{i_o}}(x^\star(p_k), \delta^\star(p_k), p_k) > (-1)\ \omega'(p_k . \varepsilon_3) + \frac{\gamma_{i_o}}{\beta} \cdot \omega'(p_k . \varepsilon_2).$$

Explicitons $\omega'(y)$:

$$\omega'(y) = \begin{cases} e^y & \text{si } y < \alpha \\ (y - \alpha)e^\alpha + e^\alpha & \text{si } y \geqslant \alpha \end{cases}$$

Comme $p_k \geqslant p_3$ et $\varepsilon_2 > \varepsilon_3$, on a $p_k\, \varepsilon_3 > \alpha$ et $p_k\, \varepsilon_2 > \alpha$

Dès lors, on obtient successivement

$$\frac{\partial \phi}{\partial \delta_{i_o}}(x^\star(p_k), \delta^\star(p_k), p_k) > -(p_k \varepsilon_3 - \alpha)e^\alpha - e^\alpha + \frac{\gamma_{i_o}}{\beta}(p_k \varepsilon_2 - \alpha)e^\alpha + \frac{\gamma_{i^\circ}}{\beta}\ e^\alpha$$

$$= -(p_k \gamma \frac{\varepsilon_2}{2\beta} - \alpha)e^\alpha - e^\alpha + \frac{\gamma_{i_o}}{\beta}(p_k \varepsilon_2 - \alpha)e^\alpha + \frac{\gamma_{i_o}}{\beta} e^\alpha$$

$$= e^\alpha(\alpha - 1) - e^\alpha \frac{\gamma_{i_o}}{\beta}(\alpha - 1) + \frac{p_k}{\beta} \varepsilon_2 (\gamma_{i_o} - \frac{\gamma}{2})$$

$$\geqslant e^\alpha(\alpha - 1).(1 - \frac{\gamma_{i_o}}{\beta}) + \frac{p_k}{\beta} \varepsilon_2 . \gamma . \frac{1}{2} \overset{\Delta}{=} \text{aux} > 0.$$

Il reste donc

$$\frac{\partial \phi}{\partial \delta_{i_o}} \quad (x^{\star}(p_k),\ \delta^{\star}(p_k),\ p_k) > \text{aux} > 0$$

ce qui contredit le caractère optimal de $(x^{\star}(p_k),\ \delta^{\star}(p_k))$ puisqu'aucune contrainte linéaire fonction de δ_{i_o} n'est saturée car $\delta^{\star}_{i_o}(p_k) > 0$.

c) Les résultats (4.16) et (4.14) démontrés respectivement en a) et b) nous donnent

$$\forall\ \varepsilon_5 > 0 \quad \exists\ p_5 \text{ tel que } \forall\ k \geqslant k_o \text{ tel que } p_k \geqslant p_5$$

$$\operatorname*{Max}_{1 \leqslant i \leqslant m+1} [\, c^r_{i,k}(x^{\star}(p_k),\ \delta^{\star}(p_k))] \leqslant \varepsilon_5$$

ce qui apporte une contradiction puisque le résultat 1 nous assure la stricte incompatibilité des systèmes traités (P_k).

□

Ce dernier résultat nous montre qu'il existe une sous-suite de la suite des indices $(k)_k$, sous-suite notée $(k_1)_1$ telle que, pour tout 1 :

$$D_{k_1} > 0$$

et donc

$$\text{Binfr}(k_1+1) = \text{Binfr}(k_1) + \beta\ c^r_{m+1,k_1}[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})]$$

Conséquence du résultat 2: Avec ces notations, la suite des valeurs numériques

$$\left[c^r_{m+1,k_1}[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})] \right]_1$$

tend vers zéro.

Démonstration

a) Remarquons que, pour tout k,

$$c^r_{m+1,k}\,(x^{\star}(p_k),\ \delta^{\star}(p_k)) \geqslant 0$$

En effet, supposons par l'absurde qu'il existe k_o tel que l'inégalité n'est pas vérifiée. Or, en vertu du caractère optimal de $[x^{\star}(p_k),\ \delta^{\star}(p_k)]$, nous avons

$$\frac{\partial \phi}{\partial \delta_i}\ [x^{\star}(p_k),\ \delta^{\star}(p_k),\ p_k] \geqslant 0 \qquad\qquad i = 1,\ \ldots,\ m$$

ce qui nous donne (raisonnement déjà fait dans la démonstration de la conséquence du résultat 1) :

$$c^r_{i,k}[x^{\star}(p_k),\ \delta^{\star}(p_k)] < c^r_{m+1,k}[x^{\star}(p_k),\ \delta^{\star}(p_k)] \qquad (4.21)$$

pour i = 1, ..., m. Donc si on suppose que $c^r_{m+1,k_o}[x^{\star}(p_{k_o}),\ \delta^{\star}(p_{k_o})]$ est négatif, (P_{k_o}) est compatible, ce qui est impossible en vertu du résultat 1.

b) La sous-suite $(k_1)_1$ a été choisie tel que $D_{k_1} > 0$ et donc

$$\text{Binfr}(k_1+1) = \text{Binfr}(k_1) + \beta.\ c^r_{m+1,k_1}\,[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})]$$

Si $\left[c^r_{m+1,k_1}[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})]\right]_1$ ne tend pas vers zéro, $(\text{Binfr}(k))_k$ qui est une suite monotone va tendre vers l'infini, ce qui est impossible puisque $(\text{Binfr}(k))_k$ est bornée par Bsupr (2) qui est fini.

□

4.6. Résultat 3

La suite des valeurs numériques $[\text{Bsupr}(k) - \text{Binfr}(k)]_k$ tend vers zéro.

Démonstration: La conséquence du résultat 1 nous assure que $[Bsupr(k) - Binfr(k)]_k$ est une suite positive décroissante. Dès lors, il suffit de trouver une sous-suite de celle-ci convergeant vers zéro et on a la thèse. Avec les notations précédentes, considérons la sous-suite

$$[Bsupr(k_1+1) - Binfr(k_1+1)]_1$$

D_{k_1} est strictement positif pour tout 1 et donc

$$Binfr(k_1+1) = Binfr(k_1) + \beta\, c^r_{m+1,k_1}[(x^{\star}(p_{k_1})\ \delta^{\star}(p_{k_1})]$$

Or, par définition de $c^r_{m+1,k_1}(x,\delta)$ au point $[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})]$, on obtient:

$$Binfr(k_1+1) = \sum_{i=1}^{m} \gamma_i\ \delta^{\star}_i(p_{k_1})$$

De plus, si l'on note

$$J = \left\{ i \mid c^r_{i,k_1}[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})] \geqslant 0 \right\}$$

il vient (# 7) que

$$Bsupr(k_1+1) \leqslant \sum_{i=1}^{m} \gamma_i\ \delta^{\star}_i(p_{k_1}) + \sum_{i\in J} \gamma_i\ c^r_{i,k_1}[x^{\star}(p_{k_1}),\delta^{\star}(p_{k_1})]$$

et donc

$$Bsupr(k_1+1) - Binfr(k_1+1) \leqslant \sum_{i\in J} \gamma_i\ c^r_{i,k_1}[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})]$$

En vertu de l'inégalité (4.21) de la démonstration précédente

$$Bsupr(k_1+1) - Binfr(k_1+1) \leqslant m\ c^r_{m+1,k_1}[x^{\star}(p_{k_1}),\ \delta^{\star}(p_{k_1})]$$

et l'on a la thèse en vertu de la conséquence du résultat 2.

□

Conséquence du résultat 3: La suite $[\text{Bsupr}(k)]_k$ converge vers R.

Démonstration: Le résultat 3 nous donne trivialement la thèse puisque, pour tout k,

$$\text{Binfr}(k) < R \leqslant \text{Bsupr}(k)$$

□

4.7. Remarques

1) A chaque mise à jour de Bsupr(k), on a un point $x^{\star}(p_k)$ vérifiant le système d'inéquations (P_k^r), système relaxé de (P) :

$$(P_k^r) \begin{cases} c_i(x) - \delta_i^k \leqslant 0 & i = 1, \ldots, m \\ l_j(x) \leqslant 0 & j = 1, \ldots, q \end{cases}$$

avec δ_i^k tels que

$$\sum_{i=1}^{m} \gamma_i \, \delta_i^k = \text{Bsupr}(k+1)$$

Or, la suite $[\text{Bsupr}(k)]_k$ converge vers R et donc la suite des relaxations

$$(\delta_1^k, \ldots, \delta_m^k)_k$$

va tendre vers la relaxation optimale au sens de notre critère.

2) L'utilisation de la constante positive "β" se justifie ainsi: choisissant cette constante petite, les mises à jour de Binfr(.) seront fréquentes mais les pas de cette mise à jour seront petits. Inversément, plus grande sera choisie la constante, plus souvent D_k sera négatif à l'étape (# 6) et parallèlement, plus intéressantes seront les mises à jour de Binfr(.) (bien que moins fréquentes).

§ 5. Conclusion

Contrairement à l'algorithme Relaxation I, cet algorithme repose sur une démonstration de convergence. Leurs résultats numériques respectifs seront présentés au chapitre V.

Chapitre V

RÉSULTATS NUMÉRIQUES

RESULTATS NUMERIQUES

§ 1. Introduction

Les résultats numériques présentés dans ce chapitre ont été obtenus sur un ordinateur DEC 2060, au moyen d'un programme utilisant deux sous-routines de minimisation (au choix de l'utilisateur) :

a) VE03AD de la bibliothèque HARWELL

b) GRSG implémentée par A. BIHAIN.

Les notations utilisées dans ce chapitre seront celles des algorithmes des chapitres I et IV.

§ 2. Résultats numériques de l'algorithme de Schnabel

2.1. Systèmes d'inéquations quelconques

Ex: 1) Nombre de variables : 2

Domaine : R^2

Sans contraintes linéaires

Contraintes non linéaires :

$$\begin{cases} c_1(x) = \sin (x_1^2 + x_2^2) + x_1 \\ c_2(x) = \frac{4}{\pi^2} (x_1 + \frac{3}{2}\pi)^2 + x_2^2 - 1 \end{cases}$$

Point de départ (4,6)

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$
0,00	8	-0,6718

Dès la première itération, on détecte la compatibilité du système par l'obtention d'un point admissible (-3,7825, -0,1269). En ce point,

$$\max_{1 \leqslant i \leqslant 2} [c_i(x^{\star}(p))] = -0,6334.$$

Ex: 2) Nombre de variables : 5

Domaine : R^5

Sans contraintes linéaires

Contraintes non linéaires :

$$\begin{cases} c_1(x) = (\frac{1}{2} x_1 - 3)\, x_1 + 2\, x_2 - 1 \\ c_2(x) = x_1 = (\frac{1}{2} x_2 - 3)\, x_2 + 2\, x_3 - 1 \\ c_3(x) = x_2 + (\frac{1}{2} x_3 - 3)\, x_3 + 2\, x_4 - 1 \\ c_4(x) = x_3 + (\frac{1}{2} x_4 - 3)\, x_4 + 2\, x_5 - 1 \\ c_5(x) = x_4 + (\frac{1}{2} x_5 - 3)\, x_5 - 1 \end{cases}$$

Point de départ (6,6,6,6,6)

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$
0,00	6	-3,6117
0,37	2	-4,1227

Le point $\begin{pmatrix} 1,5050 \\ 0,4950 \\ 0,0000 \\ 0,0179 \\ 0,5050 \end{pmatrix}$ est admissible: en ce point,

$$\max_{1 \leq i \leq 5} [c_i(x^{\star}(p))] = -0,0434.$$

2.2. Exemples concrets de la classe des problèmes présentés au chapitre II.

Ex: 1) Nombre de variables: 38

Nombre de contraintes linéaires: 1

Nombre de contraintes non linéaires: 10

Point de départ: $x_i = 100 \qquad i = 1, \ldots, 38.$

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$	$\max_{1 \leq i \leq 10} [c_i(x^{\star}(p))]$
0,005	38	-775,50	29,80
0,05	38	- 44,13	21,84
0,50	41	- 12,32	0,6811
5,00	40	- 1,124	0,2558
49,97	46	- 0,0558	0,0415
499,67	20		-0,00377

et l'on détecte la compatibilité du système par l'obtention d'un point admissible.

Ex: 2) Nombre de variables: 10

Nombre de contraintes linéaires: 1

Nombre de contraintes non linéaires: 12

Point de départ: $x_i = 10$ $\quad i = 1, \ldots, 10$

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$	$\max_{1 \leqslant i \leqslant 12} [c_i(x^{\star}(p))]$
0,20	7	-38,718	-0,304

Dès la première itération, on obtient un point admissible. Remarquons cependant que, partant du point $x_i = 0,1 \quad i = 1, \ldots, 10$; l'algorithme détecte l'incompatibilité du système. Ceci est dû au caractère non convexe des contraintes de cet exemple.

2.3. Exemples numériques du même type.

Ex: 1) Nombre de variables: 3

Nombre de contraintes linéaires: 2

Nombre de contraintes non linéaires: 4

Point de départ: (1,1,1)

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$	$\max_{1 \leqslant i \leqslant 4} [c_i(x^{\star}(p))]$
0,20	4	-0,6596	2,00
2,00	25	2,046	1,01

On détecte l'incompatibilité du système puisque $\phi^{\star}(p) > 0$.

Ex: 2) Nombre de variables: 10

Nombre de contraintes linéaires: 5

Nombre de contraintes non linéaires: 20

Point de départ: $x_i = 1, \quad i = 1, \ldots, 10$

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$	$\max_{1 \leq i \leq 20} [c_i(x^{\star}(p))]$
0,20	10	-41,42	1,589
2,00	11	- 1,186	1,170
20,00	72	-0,5064	0,111
200,00	40	1,875	0,08309

De nouveau, incompatibilité du système.

Le reste du chapitre consiste à présenter des exemples pour lesquels l'algorithme de Schnabel détecte l'incompatibilité. Dans une seconde phase, on essaie de trouver une "bonne relaxation" en employant parallèlement les algorithmes Relaxation I et Relaxation II. Le critère d'optimalité d'une relaxation utilisé est celui présenté au paragraphe 1 du chapitre 4 en prenant pour facteurs de pondération $\gamma_i = 1 \quad i = 1, \ldots, m$.

§ 3. Résultats numériques des algorithmes de relaxation (I et II)

Ex: 1) Nombre de variables: 3

Nombre de contraintes linéaires: 2

Nombre de contraintes non linéaires: 4

Point de départ: (1,1,1)

Appliquons l'algorithme de Schnabel pour détecter l'incompatibilité:

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$	$\max_{1 \leq i \leq 4} [c_i(x^{\star}(p))]$
0,20	4	-0,6595	2,00
2,00	25	2,046	1,01

La procédure de relaxation de l'algorithme Relaxation I nous donne les résultats suivants.

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	Bsupr
2,00	3	
20,00	4	
200,00	7	
200,00	8	1,03

où Bsupr est une borne supérieure sur la relaxation, c'est-à-dire Bsupr $\geq$ R défini au paragraphe 3, chapitre IV.

Comparons ces résultats avec les résultats de l'algorithme Relaxation II présentés ci-dessous:

a) avec pour paramètre β = NV : nb. de contraintes violées.

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
2,00	4 ($\phi(.,p) < 0$)			
20,00	14	+	0,8080	1,030
20,00	7	-		1,021
200,00	9	+	0,9678	1,021
200,00	13	+	1,006	1,018

et augmentant une fois de plus le paramètre p, on obtient une précision de 0,001, c'est-à-dire

$$Bsupr - Binfr \leqslant 0{,}001.$$

b) avec pour paramètre $\beta = 10.NV$

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
2,00	4 ($\phi(.,p) < 0$)			
20,00	16 ($\phi(.,p) < 0$)			
200,00	22	-		1.018
2000,00	4	+	1.016	1.017

et donc, après quatre incrémentations de p, on obtient une précision de 0,001.

Ex: 2) Nombre de variables: 6
Nombre de contraintes linéaires: 5
Nombre de contraintes non linéaires: 10
Point de départ: $x_i = 1 \quad I\ j\ 1, \ldots, 6.$

Appliquons d'abord l'algorithme de Schnabel pour finalement détecter l'incompatibilité

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$
0,05	8	négative
0,48	7	négative
4,81	143	positive

puisque $\phi^{\star}(p) > 0$. Appliquons ensuite l'algorithme de Relaxation I :

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	Bsupr
4,81	7	
48,11	46	
48,11	47	
48,11	47	
48,11	54	
48,11	54	
48,11	54	0,6975

Nous pouvons remarquer que l'emploi de l'algorithme Relaxation I devient fort coûteux dès que le système traité a plus d'inéquations.
De plus, il est moins précis que l'algorithme Relaxation II dont voici les résultats:

a) avec β = NV: nombre de contraintes violées.

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
48,11	97	+	0,5458	0,6482
48,11	11			
481,06	133	+	0,6120	0,6245
481,06	14	-		0,6245
4810,64	364	+	0,6197	0,6212

A la première itération, l'estimation "Bsupr" est déjà meilleure que celle trouvée finalement par l'algorithme Relaxation I. A la cinquième itération, la précision est déjà de 0,002.

b) avec $\beta = 10.NV$.

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
48,11	60	-		1,633
481,06	121	-		0,6228
4810,64	41	+	0,6202	0,6212

et on obtient donc, après trois itérations, une précision de 0,001.

Les exemples qui constituent le reste de ce chapitre ont plus de contraintes non linéaires et dès lors l'utilisation de l'algorithme Relaxation I serait trop coûteuse. C'est pourquoi, ne seront présentés ici que les résultats de l'algorithme Relaxation II.

Ex: 3) Nombre de variables: 6
Nombre de contraintes linéaires: 3
Nombre de contraintes non linéaires: 10
Point de départ: (1,1,1,1,1,1)

Appliquons l'algorithme de Schnabel:

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$	Max $[c_i(x^{\star}(p))]$
0,01	10	-4,475	4,027
0,12	6	-1,988	4,014
1,20	122	5,764	1,691

et l'on détecte l'incompatibilité puisque $\phi^{\star}(p) > 0$.

Utilisons l'algorithme Relaxation II:

a) avec β = NV : nombre de contraintes violées

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
1,20	110	-		3,322
12,04	58	+	2,190	3,322
12,04	56	-		3,295
120,41	50	+	3,086	3,285
120,41	120	+	3,249	3,285
1204,12	138	-		3,285

et il suffit d'une augmentation supplémentaire du paramètre p pour avoir une précision de 0,001.

b) avec β = 10.NV.

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
1,20	10 ($\phi(.,p) < 0$)			
12,04	32 ($\phi(.,p) < 0$)			
120,41	89	-		3,286
1204,12	43	+	3,268	3,285

et il suffit d'une mise à jour de Binfr supplémentaire pour avoir une précision de 0,0001.

Ex: 4) Nombre de variables: 10
Nombre de contraintes linéaires: 5
Nombre de contraintes non linéaires: 20
Point de départ: $x_i = 1$ $\quad i = 1, \ldots, 10.$

Appliquons l'algorithme de Schnabel

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	$\phi^{\star}(p)$
0,07	10	-
0,77	18	-
7,69	62	-
76,92	239	+

qui détecte l'incompatibilité du système pour p = 76,92. Appliquons ensuite la procédure de relaxation de l'algorithme Relaxation II.

a) avec β = NV: nombre des contraintes violées

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
76,92	12 ($\phi(.,p) < 0$)			
769,23	66	+	0,07759	0,0886
769,23	16 ($\phi(.,p) < 0$)			
7692,32	55	+	0,0856	0,0867
7692,32	26 ($\phi(.,p) < 0$)			
76923,17	72	+	0,08646	0,08649

On obtient une précision de 0,0001.

b) avec β = 10 NV :

Valeur de p	Nombre d'évaluations de $\phi(.,p)$	D	Binfr	Bsupr
769,23	21	-		0,1188
7692,32	107	-		0,0866
76923,17	42	+	0,08646	0,08649

et on remarque la même précision.

§ 4. Conclusion

Les résultats de l'algorithme Relaxation I nous montrent que cet algorithme n'est pas fonctionnel, mais tel n'était pas son but. Par contre, l'algorithme Relaxation II fournit des résultats intéressants puisqu'on obtient une précision assez grande après quelques mises à jour de "Binfr" et "Bsupr". Malheureusement, le paramètre de pénalisation devient vite très grand: en effet, les systèmes auxquels on applique l'algorithme de Schnabel tendent, d'une itération à l'autre, vers l'exacte compatibilité et, face à ces systèmes, l'algorithme de Schnabel doit incrémenter de plus en plus le paramètre de pénalisation. Par conséquent, face à ce type de problème, la modification de l'algorithme de Schnabel proposée au paragraphe 8 du chapitre I me semble fort intéressante. En effet, sur tous les exemples suffisamment "critiques" testés, cette modification diminue le nombre d'incrémentations du paramètre. C'est pourquoi, adapter l'algorithme Relaxation II en tenant compte de cette modification ne me semble pas être sans intérêt.